Certificate of Mailing

I hereby certify this paper is being mailed to the U.S. Patent and Trademark Office, via Express Mail No. EL648949626US on the 9 day of April, 2001.

William C. Norvell
Signature

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICANT: TRAVIS C. RILEY §
§
FILING DATE: 12/20/01 §
§
SERIAL NO. 09/745,262 §
§
TITLE: METHOD AND APPARATUS FOR §
CREATION AND TRANSMISSION OF §
FINANCIAL STATEMENT DATA §

ATTORNEY DOCKET NO.
3587/00-410

SUBSTITUTE SPECIFICATION

Respectfully submitted,

William C. Norvell

WILLIAM C. NORVELL, JR., Reg. 26,212
Beirne, Maynard & Parsons, L.L.P.
1300 Post Oak Blvd., Ste. 2500
Houston, TX 77056-3000
Telephone: (713) 960-7362

Attorney: 3587/00-410

TITLE: **METHOD AND SYSTEM FOR CREATION AND TRANSMISSION OF FINANCIAL STATEMENT DATA AND ARTICLE OF MANUFACTURE RESULTING THEREFROM**

ABSTRACT OF THE DISCLOSURE

A method, system and article of manufacture incorporates a financial institution computer system for extracting financial data within a data base, formatting the data and transmitting the formatted data via electronic mail. Electronic information on financial accounts of a customer is maintained within the data base in the financial institution computer system. The electronic information is processed to identify and extract pre-selected data and the processed data is electronically formatted for transmission to the customer via electronic mail to a location designated by the customer via electronic mail for storage within and readout on a computer system of the customer.

COPYRIGHT NOTICE

A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile, reproduction by anyone of the patent document or patent disclosure as it appears in the Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever in and to such disclosure, in whole or in part.

BACKGROUND OF THE INVENTION

(1) FIELD OF THE INVENTION: The present invention is directed to a method, an article of manufacture, and a system incorporating a computer system of a financial institution, wherein data fields are provided and access thereto is initiated for the generation of financial information for format and transmission via electronic mail to the customer on an incremental time basis.

(2) BRIEF DESCRIPTION OF THE PRIOR ART: Electronic mail (or "e-mail") is a popular way for people to communicate. Using e-mail, a person can send messages and other information such as formalized documents, etc., that are in digital form, either in the mail itself, or as an "attachment" in a rather lengthy e-mail configuration.

E-mail communication systems are generally regarded as multi-cast, store-and-forward bi-directional communication systems. A user can send e-mail messages to one or more recipients at a time. An e-mail system is regarded as bi-directional as, usually, a user can both send and receive e-mail messages. Uni-directional systems are also popular.

When using e-mail to communicate, a user will typically create a message using an e-mail program running on a computer that is or can be connected by a network to other computers. The message will include the e-mail address of the intended recipient. When the user has finished entering the message, the user will "send" the message to the intended recipient. The message is electronically transmitted via the computer network. The recipient, also using an e-mail program running on a computer connected to a computer network, can then read the received message.

A common computer network used to send and receive e-mail is the Internet. The Internet allows users to send and receive e-mail to and from computers around the world. Typically, each user will have an Internet e-mail address unique to that user, e.g., bob@pto.com. A user with an e-mail account and a computer that can connect to the Internet can easily send and receive e-mail over the Internet. Users connect to the Internet to send and receive e-mail through a number of on-line networks, such as America on Line, CompuServe, Microsoft Network, and the like. Using a computer with a modem, the user is supplied with a unique access number which is sourced through the network for sending and receiving e-mail.

In the past, banks and other similar financial organizations, have customarily sent out statements to customers on a given time basis, such as monthly. Generally speaking, separate accounts will generate statements which have been mailed to the customer. Recently, many banks and other financial institutions have offered statements in which all accounts are included in one statement, i.e., 2 or more checking accounts, a savings account, a line of credit, and the like.

Most banks offer a telephone "800" number for a customer to make specific inquiry regarding a debit, credit or balance for a particular account. Most of the "800" numbers are automated and computer generated voice responses deliver the required messages and information to the customer upon the customer entering into the phone pad identification numbers for the account and other information. Often times, these "800" numbers are not satisfactorily responsive and result in delays in communicating the desired information to the customer, with the customer being placed on "hold" for many minutes.

The present invention addresses many of the problems described above by providing a method and system for sorting, generating and presenting banking and other financial data in a format from a host computer system, i.e., a host server, to a computer integrated into an e-mail system, preferably via the Internet as above described, for a very frequent transmission of the requested and desired information to the customer, such as on a daily basis, i.e., once each morning at a designated time. In this manner, the customer may be serviced with a "daily" or almost continuous data stream containing debit, credit and balance information on a host of accounts with the financial institution, such as a Bank, savings and loan association, or the like. The information is provided in a format which offers security to the customer with respect to proper identification of accounts and the like.

DEFINITIONS

As used in the specification and the claims, the following words and phrases shall have the meanings corresponding thereto:

(1) "Checking Account": a bank account against which the depositor can draw checks.

(2) "Savings Account": an account (as in a bank) on which interest is usually paid and from which withdrawals can be made usually only by presentation of a passbook or by written authorization on a prescribed form.

(3) "Certificate of Deposit": an account (as in a bank) on which interest is usually paid at an agreed upon rate for an specific, agreed upon, time period. For example, 7% for 90 day. Withdrawals of principle are usually not allowed (without penalty) during the term of the agreement.

(4) "Loan Account": an account (as in a bank) reflecting money lent at interest.

(5) "Discount Loan Account": an account (as in a bank) reflecting money lent at interest where the interest to be charged during the term on the loan is computed at the time the money is lent and added to the amount borrowed to determine the amount of the loan. The borrower receives as proceeds, the amount of the loan less the pre-computed interest (the discounted amount);

(6) "Simple Interest Loan Account": an account (as in a bank) reflecting money lent at interest where interest charged is computed and added to the balance of the note on a daily basis using a daily interest rate factor.

(7) "Laser Notice File": a data file containing customer notices (interest paid, NSF checks, payments due, etc.) formatted for printing on a laser printed.

(8) "NSF Check": a check presented payment on an account against which the depositor can draw checks when there are not sufficient funds in the account to pay the check.

(9) "Predeterminable Time Increment Basis": For example, daily at a set time each day.

(10) "Officer Order": account information sorted in order of the initial of the bank officer assigned to the account (rather than in another order such as account number).

(11) CSV attachments: (comma separated values) information in a computer data file where each file record contains several data elements each separated by a comma. The comma marks the end of one data element and the beginning of another.

(12) Qwicken attachments: Information in a computer data file formatted to the requirements of the popular personal accounting program Quicken, for importing financial information into the program.

SUMMARY OF THE INVENTION

The present invention provides a method, system and article of manufacture incorporating a financial institution computer system for extracting financial data within a data base in the computer system and for formatting the data and thereafter transmitting the formatted data via electronic mail. Electronic information is maintained on financial accounts of a customer within the data base in the financial institution computer system. The electronic information is processed within the data base to identify and extract pre-selected data therefrom. The data is electronically formatted for transmission to the customer via electronic mail. The formatted data is transmitted to a location designated by the customer via electronic mail for storage within and readout on a computer system of the customer.

The financial institution preferably is a bank. As used herein, the computer system may include a computer server, one or more personal computers and any other electronic computer system well known to those skilled in the art and utilized in financial institutions for the maintaining of electronic information pertaining to checking, savings, certificates of deposit, loan accounts, and the like. Electronic information is processed to identify and extract pre-selected data therefrom, such as mini trial balances for checking accounts, savings accounts, certificates of deposit, loan accounts, and discount loan accounts, simple interest loan accounts, and any and all other special service accounts of the financial institution for each of its customers. The information is electronically formatted such as by use of a "WINDOWS" type computer operating system including menu selections for customers, options, verify files, enable auto e-mail, generate e-mail, broadcast, help and exit.

A customer setup configuration is generated through sorting of the data field in the computer for each customer in at least one of the following fields sorted by file within the computer: account number; reference number; account name; account type; balance for designated accounts; transactions for designated accounts; non-sufficient funds designation; and day or date of report transmission.

DESCRIPTION OF THE ILLUSTRATIONS

Fig. 1 is a printout of a sample or representative e-statement illustrating the preferred layout and configuration of the report generated in accordance with the present invention.

Fig. 2 is a view of the main menu for generating the e-mail report of the preferred embodiment of the invention in WINDOWS® format as it would appear on a personal computer CRT or screen.

Fig. 3, similar to the illustration of Fig. 2, is an illustration of the customer's selection, 39, from the main menu.

Fig. 4, is an illustration of the view of the customer set-up generated by selection at the add/edit button 56.

Fig. 5 is a view similar to that of Fig. 4 showing the customer account set up configuration which will appear by activating the accounts button 68 shown in Fig. 4.

Fig. 6 is a view similar to the previous figures illustrating a sub-menu option of "change report path" and "change loan data path" by clicking onto the options field 40 illustrated in Fig. 2.

Fig. 7 is a view similar to the other figures of the sub-menu option for "change system information" resulting from scrolling at 107 in the sub-menu option illustrated in Fig. 6.

Fig. 8 is a view similar to the previous views of the sub-menu option of "Edit Auto E-Mail Settings" by scrolling at 107 in the sub-menu illustrated in Fig. 6.

Fig. 9 is yet another sub-menu option which may be position by again scrolling at 107 on the sub-menu profile of Fig. 6.

Fig. 10 is a view of yet another sub-menu option of "Edit Transaction Codes" which may be selected by again scrolling at 107.

Fig. 11 is an illustration of the selection on the main menu of "enable auto e-mail" and "disable auto e-mail", field 42 as illustrated in Fig. 2.

Fig. 12 is yet another illustration of another main menu selection in the "broadcast" field 44 of Fig. 2. Fig. 13 illustrates the method of the generation of the e-statement in logic and step format.

Fig. 14 is a view similar to Fig. 13 illustrating the sub-program of "make statements".

Fig. 15 is a view similar to Figs. 13 and 14 showing another sub-program pertaining to reading of the loans files.

Fig. 16 is a view similar to Figs. 13 through 15 showing the logic and step path for the reading of deposit trial balances files.

Fig. 17 is yet another logic/step view similar to Figs. 13 through 16 of the sub-program of reading transactions and NSF (non-sufficient funds) files.

Figs. 18A through 18D together constitute an illustration of a representative printout of a detailed statement generated in accordance with the present invention, and as built from data input as illustrated in Figs. 2 through 12 in accordance with the description in the specification relating thereto.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Now with reference to Fig. 1, there is shown a sample of an e-statement preferred format 1 which is transmitted electronically from the financial source, through a computer via conventional e-mail Internet services.

The statement 1 may appear upon a computer screen, or the like, and/or may be printed out in any tangible format. As shown, the e-statement 1 contains a first line 2 containing the title and the source of the statement and any advertising or promotional or legal notice requirements, such as “Member FDIC”. A date line 3 is provided for specifying the statement generation date and time of generation. As shown, the statement 1 has a statement generation field 3 specifying that it is generated “as of close of business Monday Oct. 16, 2000". A salutation/description line 4 brings the attention of the reader to the subject matter of the e-statement 1.

Fields 5, 6 and 7 are clustered on one line to identify the type of account, and the digitized account number 6, which may have one or more sub-fields in “x”ed or other disguising configuration, for security purposes. Field 7, as shown, describes the account as “Household Account”.

Below the field lines 5, 6 and 7 are found the Available Balance field 8 and the Exact Balance field 9 as of the date of the generation of the e-statement 1, reflected in field 3.

A field 10 is identified as “Last Deposit”, giving the amount field 11 in numbers, and an “as of” or “on” field 12 for identifying the date of the last deposit 10 which may or may not, be the same date as the generation date 3 of the e-statement 1.

Field 13 specifies the transactions for the account identified in field 6 by date 14, serial or transaction number 15, amount 16 and transaction source and type 17. As shown, under the date field 14, three transactions, 18, 19 and 20 are shown for Oct. 16, 2000 with each being identified by a separate serial number 15. Each of the amounts 16 for the transactions 18, 19 and 20 are identified and described in the transaction identification field 17, such as a check, ACH, debit or a VISA® banking credit or debit card, well known to those skilled in the art.

The e-statement 1 also has a similar information in fields for another account 22 which is identified in field 1 as a "savings account" and further identified in description field 23 as a "Household Savings Account". An available balance field 24 similar to that in field 8 is shown for the savings account 21 in a specific monetary amount at field 25. A last deposit field 26 indicates, as shown, an amount of $2,000 in field 27 deposited to the account on Oct. 13, 2000 as shown in field 28.

Again, particular transactions for the account are shown in field 26 with the account being respecified in field 27. The transactions are identified along field line 29 by date, field 30, serial number, field 31, amount, field 32 and transaction source and type, field 33. The date is provided in field 34 with the serial number identification field 35 and the specified amount of the transaction in the amount field 36. In field 37, the transaction is identified as a "withdrawal", the amount of $40.00, field 36.

The e-statement 1 may be provided in a number of formats, with figure 1 being only representative of an arrangement of the accumulation of the financial data in the statement format 1.

The operation of the e-statement method and program will now be described. Figures 2 through 12 illustrate various computer generated video screen or CRT displays of various menu selections incorporated within a preferred program for the e-statement process. As shown, and as previously described, the program is preferably generated through the computer by means of a WINDOWS® computer operating system which is well known and generally publically available in numerous versions.

With first reference to Fig. 2, a main menu 38 is displayed. The main menu has sub-menus identified as customers 39, options 40, verify files 41, enable auto e-mail 42, generate e-mail 43, broadcast 44 and general help fields or sub-menus 45 and exit field 46.

The customer's menu or field 39 is more particularized in Fig. 3 and discussed hereafter in detail. In this customer's field 39, a customer's name, e-mail address and account information are entered or changed, from time to time.

The main menu 38 also contains an option field 40 for the purpose of changing certain program options such as the directory path to reports, default fonts and graphics for statements, auto e-mail settings, and to define core processor transaction codes, and the like.

The verify files field 41 assures verification that all files which are needed to generate a current days e-statements have been downloaded from the financial source core computer processor.

The enable auto e-mail field 42 is used to turn on or off any program feature monitoring the progress of the daily report downloads from the financial organization's core computer processor. Additionally, this field 42 generates e-statements as soon as all required reports have been received and processed just prior to generation of the e-statement configuration to the customer.

The generate e-mail field 43 is utilized to generate a current day's, week's or months e-statements which may be prepared and sent on any incremental time basis. This field 43 is a manual request field. The program itself verifies to confirm that all required report files are available. However, if some required reports are not available or are incomplete, the program allows the user the option to continue or abort the request through activation of the generate e-mail field 43 as shown and described in more detail in Fig. 9 and discussion relating thereto.

The broadcast field 44 is utilized to send an e-mail message to all or a selected group of customers receiving e-statements to announce changes, delays, modifications, or any other material information which is desired to be generated and transmitted to a select number of customers.

The help menu selection field 45 is conventional in nature and is provided for purposes of immediate computer program operating assistance.

Finally, the exit program or field 46 may be utilized for click-on to return to the original WINDOWS® desktop configuration after termination of entry into the program.

Now with particular reference to Fig. 3, the customer's sub-program or field 39 in the main menu 38 will now be described. As shown in Fig. 3, the customer's field 39 is shown in spreadsheet configuration. Fields 50 through 54 are entered in negative/zero configuration where the negative is reflective of a "true" state and a "0" is indicative of a false state, as further described below. Sub-fields 47 through 55 are displayed across the spreadsheet in a horizontal configuration. The "Id" field permits entry of a unique number or code correlated to a single customer. Within the customer name field 48 are listed the individual customer names, by individual or business. The e-mail address field is horizontally displayed adjacent the customer name field 48 which is followed by fields 50-54 for a specific service information. For example, "CHRG" field 50 is a field indicating whether or not the customer is to be charged for the service of providing the e-statement. A negative number in this field 50 would indicate that the customer is to be charged for the e-mail statement service. The "CONF" field 51 is utilized to verify that the customer has returned the confirmation of the e-mail address. The "SUSP" field 52 is utilized to indicate possible temporary or permanent suspension of the service for this particular customer. The "No Ad" field 53 is utilized to indicate whether or not electronic advertising is to accompany the e-statement with transmission to this particular customer. The "Rate" field 54 is utilized to indicate the transmission of various interest rates being paid to customers on deposit accounts or charged to customer on loan accounts by the bank.

The "Add/Edit" button 56 is a click field which will allow the program user to add a new customer, to delete a customer, or change any of the information about the customer in any of the fields 47 through 55. Selecting this button 56 with a customer's information highlighted will display that customer's information for editing purposes. Selecting this button without a customer's information highlighted will display the first customer's information for editing purposes, or , alternatively, the last customer's information for such purposes, as described below. The "Close" button 57 will, of course, return the program to the main menu display, as in Fig. 2.

The Add/Edit button 56 window display is particularized in Fig 4. Fields 47 through 54 are displayed vertically and correspond to the horizontal configuration for such fields in Fig. 3. The address name field 48 may be filled in by a click-on and type in of the data in field 48A. Likewise, the e-mail address field 49 may be clicked on to insert such information in field 49A. As indicated, charge and confirmed fields 50 and 51 are positively indicated by click-on at fields 58 and 59. Fields 60, 61 and 62 likewise are that they are click-checked or the field left blank, as the case requires. The sort name field 63 is filled in at corresponding field 64 by use of first, last or code names, as required. The Add button 65 is conventional and clears all the fields and sets them to their default value thus allowing the entry of a new customer's information, or update of such information. Likewise, the delete button 66 deletes the displayed customer from the field. When the delete button 66 is utilized, no additional e-statement will be generated for that particular customer. The update button 67 is utilized to update the customer's information with the contents displayed on the screen, Fig. 3, Fig. 4, or adds a new customer to the database after the information has been entered. The Account button 68 allows the program user to add or edit account information for the customer currently being displayed. The Close button 69 closes the window display Fig. 4 to return to the display format of Fig. 3.

Now with reference to Fig. 5, there is shown the statement account display which allows the program user to add or edit account information for the customer currently being displayed in particular fields. The Account field 70 is entered in space 71 in alpha numeric format from the financial systems main computer processing unit. This number is assigned at the time the account is opened, such as a purchase of a certificate of deposit, or completion of a loan transaction. A limited number of digits for the account number field 71 are permitted to be displayed on the e-statement or account attachments as illustrated in Fig. 1, for security purposes. The reference number field 72 is entered in space 73 from the financial organization's main computer and this number may, or may not, be the same as the account number 71, depending upon the particular operations of the bank"s central computer processing system. However, the number in the field 73 is never displayed or printed on the e-mail statements or account statement attachment due to a computer block for printout of this particular number, also for security purposes.

The account name field 74 is filled in at space 75 as it will appear on the customer's e-statement, as in Fig. 1. This can be any name which is meaningful to the customer. For security reasons it should not be the customer's actual account name and preferably will be, as reflected in the e-statement of Fig. 1, identified as "Household Account" or "Regular Checking", or the like. The customer will be able to identify the respective accounts by the last four digits of the account number, as printed, and not particularly with reference to the specific name in the field 75. The account type field 76 is entered at 77 by scrolling on button 78 for one of a number of account types reported on the program, such as checking, savings, certificate of deposit, commercial loan, consumer loan, line of credit, revolving account, or the like.

The reports balance field 79 is checked in the adjacent area if the balance of the account is to be reported each day, week, month, or the like, along with certain other information specific to each account type.

The scroll down button 78 may be clicked on to select one of a number of account type 76 in field 77, such as checking accounts, savings accounts, loans, certificates of deposits, and the like.

If the report balance field 79 is checked in the appropriate location, supplemental information for the various types of accounts as indicated in the field 77 will be generated and included within the e-statement. For example, if checking accounts is scrolled into the field 77 by click-on to the arrow button 78, current balance, available balance, date of last deposit, amount of last deposit, posted transactions and NSF (None Sufficient Funds) Items will be generated. If "Loans" is scrolled into the field 77, current balance, interest rate, payment amount, last payment due, next due date, maturity date, payoff balance, interest year-to-year and collateral description information will also be generated. If savings accounts is scrolled into the field 77, available balance, date last deposit, amount of last deposit and posted transactions will be generated into the e-statement automatically. If certificates of deposit are entered into the field 77, current balance, next payment date and accrued interest information will be generated.

If the report transactions field 80 is checked, all transactions posted to the account will be listed in the e-statement along with other information, as described above.

If the report NSF field 81 is checked, any debit, such as a check, or the like, presented for payment when the account does not have sufficient funds to pay the debit will be reported, so that the customer may make provision for supplemental or immediate deposits, or the like or transfer of other funds into the account. NSF Items appear before any other account information on the e-statements as an alert courtesy to the customer. This option is, of course, valid for only checking accounts or accounts similar to conventional checking accounts.

The attached statement field 82 is checked if it is desired for any statement produced for the account by the bank's core processing computer system to be added to the e-statement as an attachment. Integrated statements, i.e., those with more than one account on a single statement, may be attached only once, if desired. The extract ASCII field 83 is checked if it is desired to generate an ASCII text file each day that there are transactions for an account and attached to that day's e-statement. This file can be imported into many different accounting programs for account reconciliation operations, as desired.

The extract Qwicken™ field 84 is checked if a file in Qwicken™ format is to be created each day there are transactions for an account and attached to that day's e-statement for transmission to the customer. This file can be imported into Qwicken™ for account reconciliation. The report on given days, such as Monday through Friday's fields 85 through 89 are checked for generation of e-statement for that particular day. For example, the customer may only want certificate of deposit or loan information on a weekly basis since these accounts customarily have limited transactions, whereas checking accounts may have transactions on them several times each day. In such case, a report may be generated for each day of the week by checking in the appropriate fields 85 through 89. Alternatively, field 90 may be checked if a report is to be generated only on a monthly basis and field 91 is completed to indicate on the day of such month that the e-statement is to be generated and transmitted to the customer.

Field 98 is the "Last Statement" field and appears at the upper right hand corner of the window. This field is the date of the last statement generated for the account and is utilized for information purposes only. Add, delete, update and previous and next buttons 92 through 96 are provided for respective adding, deleting, updating, or moving to previous and next displays, in conventional format. Likewise, the Close button 97 is provided to close the window and return to the previous format.

Fig. 6 represents the visual configuration of the program appearing on the CRT or other screen from the main menu selection of options 40. Fig. 6 is illustrated with the change report path and change loan data path submenu options preparing in the window. These options allow the program user to designate the full path name to the directory where the computer download financial report files from the core computer are downloaded at the end of each day, the like. As shown on Fig. 6, the correct report download directory may be selected by either clicking on to a location field 106 or by entry of the file name 99 in field 100 or by scrolling at 107 for designation in the field 104 of the files of type 103 and then clicking to the open button 101. The selection may be cancelled by clicking at 102 prior to opening the file at 101. Opening the file 101 will change the path to the files in the windows registry. The report path and the loan data path typically will be identical.

As shown in Fig. 7, another submenu option is the "change system information" option. This submenu option permits the program user to change some of the program options, as provided. The system name field 108 will permit entry at location 109 of the brand name used for the title of the statement, such as "E-Statement". Other service mark titles may be utilized as desired. The name which is entered and placed in the field 109 will be used on all customer e-mail that is generated.

The bank name field 110 identifies the supplier of the service at 111 and is entered in the e-statement in the "from" field (see Fig. 1). The statement font size

112 is entered in space 113 which will be the font size used in the account statements created as attachments.

The statement graphic field 114 is inserted at 115 if a valid graphic file name is to be entered and the graphic will be added to the upper left hand corner of the statement attachments, to include a logo, or the like in the e-statement configuration transmitted to the customer. The brouse button 116 is used to select reference to a graphic file, which may be created as needed. The save button 117 may be used to save the currently displayed program settings and close the window to return to the main menu. Likewise, the close button 118 will close the window and return to the main menu without saving any changes.

As shown in Fig. 8, another submenu option is the "Edit Auto E-Mail Settings". This submenu option will allow the program user to change the time to begin the e-statement generation each day, and a number of minutes between attempts to automatically generate e-statement through the computers and the internet to receipt by the customer, if all reports needed are not available at the time of initial desired generation.

The "begin e-mail" function 120 time is entered in space 121 by scrolling up or down on buttons 122, 123. This time is the time that the computer is instructed to begin trying to generate and send the day's e-statement. The minutes between attempts field 124 is selected and entered at 125 by scroll up or down at buttons 126 and 127 to reflect the time in between attempts to try to generate such e-statement because downloads from a core data processing system may take several minutes or even hours. If all of the reports are not completely downloaded, continual repetitious checks for the reports may not be successful. Therefore, by increasing the time between checks, a computer processing usage can be limited and e-statement may be generated in a reasonable time after the last report is received.

Save and cancel button 128 and 129 are utilized to either save the currently displayed program setting and close the window and return to the main menu or to close the window and return to the main menu without saving such information.

The program contemplates and enable/disable submenu option, which is used only during testing. Disabling users will check the "Suspend" option for each e-statement customer except the customer whose customer ID is a specific number, such as "10". This will allow the program user to test new program settings, while sending e-statements to only one customer, such as a staff member of the financial organization. Selecting "enable" will return all customer's to their previous suspended status.

The next submenu option is illustrated in Fig. 9 and provides the set up and screen profile for the generation of the selected e-statement. This submenu option allows the user to generate the selected portions of the daily e-statements generated by the financial organization main computer as well as to attach a personal message. This submenu may be used to send information of a special nature that may not have been available at the time that the e-statements were generated on a daily, weekly or other basis. Fields 130 through 135 may be checked by appropriate click for balance information, transactions, loan information, statement attachments, CSV attachments or Qwicken™ attachments. Field 136 may be used for personal messages. After completion of this option, the continue button is click at 37 or the operation may be cancelled by clicking at 138.

Fig. 10 illustrates another submenu option generally referred to as “edit tran codes”. This submenu allows the operator to provide descriptions used in the e-statement for the different type of transactions posted to deposit and/or loan accounts. A transaction code field 139 is typed in in area 140 and corresponds to a pre-determined code in the computer for a given transaction type and is assigned by the main data processor. A description field 140 is used for insertion of an identification if the title of the transaction code, such as “new account opening deposit” at area 141. A debit field 142 may be checked at location 143 to indicate that the transaction code is for a debit transaction, such as a check or an ACH debit for insurance, car payment or the like. The add button 144 clears the content of all the fields and resets them to their default value enabling the program user to enter a new transaction code, when desired. Likewise, the delete button 145 will delete the information for the currently displayed transaction code. The refresh button 146 is used to realign all the transaction codes in numerical order. This button may be used after a new transaction code has been entered to get it in the proper numeric sequence for viewing. The update code 147 is used to update information with the contents displayed on the screen or may be used to add a new transaction code to the database after the transaction code’s information has been entered.

The close button148 will close the window and return to the previous screen. Arrow keys 149 and 150 will function as "next" and "previous" buttons for displaying or scrolling from one transaction code to the next as reflected in field 151.

Now returning to referred Fig. 2, the verify files field 41 is used to verify that all necessary files are available to produce the day's e-statements. Each of these files is given a specific code and they're contained within the main or core computer system. Typically, and preferably, these files will include the following:

1. Daily checking account mini-trial balance.
2. Daily savings account mini-trial balance.
3. Daily certificate of deposit account mini-trial balance.
4. Daily loan account mini-trial balance.
5. Daily discount loan trial balance.
6. Daily simple interest loan trial balance.
7. Daily customer's without account trial balance.
8. Daily posting journal containing all posted transactions for all account types in account number order.
9. Daily posting journal containing posted transactions for all account types in amount order.
10. Daily file containing all statements printed the previous.
11. A laser notice file.
12. The daily NSF checks notices in officer order (including copy).

If all the files are not present in the downloaded director, a message to that affect will be displayed on the screen. If they are not, a message showing each missing file is displayed as it is check. Several of the files listed above are not used for information but the presence of the file indicates completion of certain downloading steps.

Fig. 11 is a main menu selection display for enablement and disablement of automatic e-mail generation. This function eliminates a need for a program user to remember to generate the e-statement each day at a certain time. When the program is running and auto e-mail is enable, the program monitors the time of day. When the selected time is reached, such as reflected in the next schedule field 152 reflected in area 153, it checks to see if all reports needed have been downloaded. If they have, the program generates the daily e-statements and waits another 24 hours, or other time designed period, before repeating the operation. If not, the program will repeat checking every few minutes or other time increments until all the files have been downloaded. During the waiting phase, the display is as shown in Fig. 11. The last complete field 150 shows a date and time in the area 151 for the last completed cycle. The next schedule field 152 is completed in area 153 to show beginning of the next cycle for the generation of the e-statements. The last attempt field 154 will automatically reflect in area 155 a “complete” or “waiting” indicator. The missing files area 156 will automatically reflect in location 157 the number or identification of files that are missing and are required for the complete generation of the e-mail statement. Finally, current time in field 158 is reflected in area 159.

Fig. 12 reflects the view on the screen of the main menu selection for "broadcast". The broadcast field is identified as "send to" at 160 and a list of selected classification of users, such as "all paying users" may be reflected and selected in area 161 by scrolling on 162. The subject of the broadcast is identified at field 163 in area 164, such as "monthly charges". A message describing the subject is reflected in the field 164 may be manually inserted in a message are 165 and sent to the selected grouping of customers designated in 161 by clicking on the send 166. Alternatively, the message and the broadcast maybe cancelled by clicking at 167 which will return the user to the main menu.

The software preferably utilized to implement the present invention may be any one of a comparatively low level machine code, such as visual basic. The logic and sub-routines utilized to form the e-statement method disclosed in Figs. 1 through 14 is set forth below:

```
frm About - 1
Option Explicit
' Reg Key Security Options...
Const KEY_ALL_ACCESS = &H2003F


' Reg Key ROOT Types...
Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1                    ' Unicode nul terminated string
Const REG_DWORD = 4                    ' 32-bit number


Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"


Private Declare Function RegOpenKeyEx Lib "advapi32" Alias "RegOpenKeyExA"
(ByVal hKe
y As Long, ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As
Lon
g, ByRef phkResult As Long) As Long
Private Declare Function RegQueryValueEx Lib "advapi32" Alias "RegQueryValueExA"
(ByV
al hKey As Long, ByVal lpValueName As String, ByVal lpReserved As Long, ByRef
lpType
As Long, ByVal lpData As String, ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long) As Long


Private Sub Form_Load()
  Me.Caption = "About" & " " & SystemName
  lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
  lblTitle.Caption = App.Title
```

```
End Sub



Private Sub cmdSysInfo_Click()
  Call StartSysInfo
End Sub


Private Sub cmdOK_Click()
  Unload Me
End Sub


Public Sub StartSysInfo()
 On Error GoTo SysInfoErr


  Dim rc As Long
  Dim SysInfoPath As String


  ' Try To Get System Info Program Path\Name From Registry...
        If    GetKeyValue(HKEY_LOCAL_MACHINE,    gREGKEYSYSINFO,
gREGVALSYSINFO, SysInfoPath) T
hen
  ' Try To Get System Info Program Path Only From Registry...
```

```
        ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC,
gREGVALSYSINFOLOC, SysI
nfoPath) Then
        ' Validate Existence Of Known 32 Bit File Version
        If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
            SysInfoPath = SysInfoPath & "\MSINFO32.EXE"

        ' Error - File Can Not Be Found...
        Else
            GoTo SysInfoErr
        End If
    ' Error - Registry Entry Can Not Be Found...
    Else
        GoTo SysInfoErr
    End If

frmAbout - 2

    Call Shell(SysInfoPath, vbNormalFocus)

    Exit Sub
SysInfoErr:
    MsgBox "System Information Is Unavailable At This Time", vbOKOnly
End Sub


Public Function GetKeyValue(KeyRoot As Long, KeyName As String, SubKeyRef As
String,
ByRef KeyVal As String) As Boolean
    Dim i As Long                                   ' Loop Counter
    Dim rc As Long                                  ' Return Code
    Dim hKey As Long                                  ' Handle To An Open Regis
try Key
    Dim hDepth As Long                                '
    Dim KeyValType As Long                               ' Data Type Of A Registry
 Key
    Dim tmpVal As String                              ' Tempory Storage For A R
egistry Key Value
    Dim KeyValSize As Long                              ' Size Of Registry Key Va
riable
    '------------------------------------------------------------
    ' Open RegKey Under KeyRoot {HKEY_LOCAL_MACHINE...}
    '------------------------------------------------------------
     rc = RegOpenKeyEx(KeyRoot, KeyName, 0, KEY_ALL_ACCESS, hKey) ' Open
Registry Key
```

```
    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError        ' Handle Error...


    tmpVal = String$(1024, 0)                         ' Allocate Variable Space

    KeyValSize = 1024                                 ' Mark Variable Size


    '------------------------------------------------

    ' Retrieve Registry Key Value...

    '------------------------------------------------

    rc = RegQueryValueEx(hKey, SubKeyRef, 0, KeyValType, tmpVal, KeyValSize)   '
Get
/Create Key Value


    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError        ' Handle Errors


    tmpVal = VBA.Left(tmpVal, InStr(tmpVal, VBA.Chr(0)) - 1)

    '------------------------------------------------

    ' Determine Key Value Type For Conversion...

    '------------------------------------------------

    Select Case KeyValType                              ' Search Data Types...

    Case REG_SZ                                       ' String Registry Key Dat
a Type

        KeyVal = tmpVal                               ' Copy String Value

    Case REG_DWORD                                      ' Double Word Registry Ke
```

```
y Data Type
     For i = Len(tmpVal) To 1 Step -1                  ' Convert Each Bit
        KeyVal = KeyVal + Hex(Asc(Mid(tmpVal, i, 1)))  ' Build Value Char. By Ch
ar.
     Next
     KeyVal = Format$("&h" + KeyVal)                   ' Convert Double Word To
String
   End Select

   GetKeyValue = True                                  ' Return Success
   rc = RegCloseKey(hKey)                              ' Close Registry Key
   Exit Function                                 ' Exit

GetKeyError:   ' Cleanup After An Error Has Occured...
   KeyVal = ""                                   ' Set Return Val To Empty
 String
   GetKeyValue = False                                 ' Return Failure

frmAbout - 3


   rc = RegCloseKey(hKey)                              ' Close Registry Key
End Function
```

```
VERSION 5.00
Begin VB.Form frmAbout
   BorderStyle    =  3  'Fixed Dialog
   Caption        =  "About eStatement"
   ClientHeight   =  3630
   ClientLeft     =  45
   ClientTop      =  330
   ClientWidth    =  5865
   ClipControls   =  0   'False
   Icon           =  (Icon)
   LinkTopic      =  "Form1"
   MaxButton      =  0   'False
   MinButton      =  0   'False
   ScaleHeight    =  3630
   ScaleWidth     =  5865
   ShowInTaskbar  =  0   'False
   StartUpPosition =  1  'CenterOwner
   Tag            =  "About Project1"
   Begin VB.PictureBox picIcon
      AutoSize       =  -1  'True
   BackColor     =  &H00C0C0C0&
   ClipControls  =  0  'False
   Height        =  1260
   Left          =  60
   Picture       =  (Bitmap)
   ScaleHeight   =  1200
   ScaleMode     =  0  'User
   ScaleWidth    =  1200
   TabIndex      =  2
   TabStop       =  0  'False
   Top           =  0
   Width         =  1260
End
Begin VB.CommandButton cmdOK
   Cancel        =  -1  'True
   Caption       =  "OK"
   Default       =  -1  'True
   Height        =  345
   Left          =  4245
   TabIndex      =  0
   Tag           =  "OK"
   Top           =  2625
   Width         =  1467
```

```
End
Begin VB.CommandButton cmdSysInfo
  Caption        =  "&System Info..."
  Height         =  345
  Left           =  4260
  TabIndex       =  1
  Tag            =  "&System Info..."
  Top            =  3075
  Width          =  1452
End
Begin VB.Label lblDescription
  Caption        =  <...>
  ForeColor      =  &H00000000&
  Height         =  1170
  Left           =  1380
  TabIndex       =  6
  Tag            =  "App Description"
  Top            =  1125
  Width          =  4005
End
Begin VB.Label lblTitle
```

```
      Alignment     =  2  'Center
      Caption       =  "eStatement"
      BeginProperty Font
         Name          =  "Lucida Calligraphy"
         Size          =  24
         Charset       =  0
         Weight        =  400
         Underline     =  0   'False
         Italic        =  0   'False
         Strikethrough =  0   'False
      EndProperty
      ForeColor     =  &H00000000&
      Height        =  480
      Left          =  1230
      TabIndex      =  5
      Tag           =  "Application Title"
      Top           =  240
      Width         =  4095
   End
   Begin VB.Line Line1
```

```
  BorderColor   =  &H00808080&
  BorderStyle   =  6  'Inside Solid
  Index        =  1
  X1          =  225
  X2          =  5657
  Y1          =  2430
  Y2          =  2430
End
Begin VB.Line Line1
  BorderColor   =  &H00FFFFFF&
  BorderWidth   =  2
  Index        =  0
  X1          =  240
  X2          =  5657
  Y1          =  2445
  Y2          =  2445
End
Begin VB.Label lblVersion
  Alignment     =  2  'Center
  Caption      =  "Version"
  Height       =  225
  Left        =  1260
  TabIndex      =  4
```

```
    Tag          =  "Version"
    Top          =  780
    Width        =  4095
  End
  Begin VB.Label lblDisclaimer
    Caption      =  <...>
    ForeColor    =  &H00000000&
    Height       =  825
    Left         =  255
    TabIndex     =  3
    Tag          =  "Warning: ..."
    Top          =  2625
    Width        =  3870
  End
End
```

frmAccounts - 1

```
Option Explicit
Dim Ds As Recordset, SQL As String
Sub ClearFields()
  txtFields(0) = ""
txtFields(1) = ""
txtFields(2) = ""
txtFields(3) = ""
Combo1.ListIndex = 0
chkFields(3) = 1
chkFields(4) = 1
chkFields(5) = 1
chkFields(6) = 1
chkFields(1) = 0
chkFields(0) = 0
chkDays(1) = 1
chkDays(2) = 1
chkDays(3) = 1
chkDays(4) = 1
chkDays(5) = 1
chkDays(6) = 0
txtFields(4) = ""
lblLabels(11) = ""
Screen.MousePointer = 0

End Sub
Sub LoadData()
With tblAccounts
```

```
txtFields(0) = ![User ID]

txtFields(1) = ![Account Number]

txtFields(2) = ![Reference Number]

txtFields(3) = ![Name]

If Trim$(![Account Type]) = "Checking" Then Combo1.ListIndex = 0

If Trim$(![Account Type]) = "Savings" Then Combo1.ListIndex = 1

If Trim$(![Account Type]) = "CD" Then Combo1.ListIndex = 2

If Trim$(![Account Type]) = "Loan" Then Combo1.ListIndex = 3

chkFields(3) = Abs(![Report Balance])

chkFields(4) = Abs(![Report Transactions])

chkFields(5) = Abs(![Report NSF] = True)

chkFields(6) = Abs(![Attach Statement])

chkFields(1) = Abs(![Extract ASCII])

chkFields(0) = Abs(![Extract Qwicken])

chkDays(1) = Abs(!Monday)

chkDays(2) = Abs(!Tuesday)

chkDays(3) = Abs(!Wednesday)

chkDays(4) = Abs(!Thursday)

chkDays(5) = Abs(!Friday)

chkDays(6) = Abs(![Monthly Only])

txtFields(4) = ![Monthly Day] & ""

lblLabels(11) = Format$(![Last Report], "mm/dd/yyyy")
```

```
End With
 Screen.MousePointer = 0
 On Error Resume Next
 txtFields(1).SetFocus
 On Error GoTo 0
End Sub


Private Sub cmdAdd_Click()
 ClearFields
 txtFields(0) = CurrentID
 txtFields(1).SetFocus
End Sub


Private Sub cmdDelete_Click()
```

```
 If MsgBox("Delete this account?", vbYesNo + vbQuestion, "DELETE ACCOUNT") <>
vbYes
Then Exit Sub
 tblAccounts.Seek "=", txtFields(0), txtFields(1)
 If tblAccounts.NoMatch = False Then
   tblAccounts.Delete
   ClearFields
   tblAccounts.Seek ">=", CurrentID, 0
   If tblAccounts![User ID] = CurrentID Then
     LoadData
   End If
 End If
End Sub

Private Sub cmdNExt_Click()
On Error Resume Next
tblAccounts.MoveNext
If Err > 0 Or tblAccounts.EOF Then tblAccounts.Seek ">=", CurrentID, 0: If tblAccount
s.EOF Then Exit Sub
If tblAccounts![User ID] > CurrentID Then tblAccounts.Seek ">=", CurrentID, 0
If tblAccounts![User ID] = CurrentID Then
   ClearFields
   LoadData
End If
End Sub

Private Sub cmdPrevious_Click()
tblAccounts.MovePrevious
If Err > 0 Or tblAccounts.BOF Then tblAccounts.Seek "<=", CurrentID, 9999999999#
If tblAccounts![User ID] > CurrentID Then tblAccounts.Seek "<=", CurrentID, 999999999
9#
If tblAccounts![User ID] = CurrentID Then
   ClearFields
   LoadData
End If

End Sub

Private Sub cmdUpdate_Click()
 tblAccounts.Seek "=", txtFields(0), txtFields(1)
 If tblAccounts.NoMatch Then tblAccounts.AddNew Else tblAccounts.Edit
 With tblAccounts
  ![User ID] = txtFields(0)
 ![Account Number] = txtFields(1)
 ![Reference Number] = txtFields(2)
 ![Name] = txtFields(3)
 ![Account Type] = Combo1
 ![Report Balance] = (chkFields(3) = 1)
 ![Report Transactions] = (chkFields(4) = 1)
 ![Report NSF] = (chkFields(5) = 1)
 ![Attach Statement] = (chkFields(6) = 1)
 ![Extract ASCII] = (chkFields(1) = 1)
 ![Extract Qwicken] = (chkFields(0) = 1)
 ![Monday] = (chkDays(1) = 1)
 ![Tuesday] = (chkDays(2) = 1)
 ![Wednesday] = (chkDays(3) = 1)
 ![Thursday] = (chkDays(4) = 1)
 ![Friday] = (chkDays(5) = 1)
 ![Monthly Only] = (chkDays(6) = 1)
 If ![Monthly Only] Then
   ![Monthly Day] = Val(txtFields(4))
 Else
   ![Monthly Day] = 0
 End If
End With
```

```
 tblAccounts.Update
 Screen.MousePointer = 0
 txtFields(1).SetFocus

End Sub

Private Sub cmdClose_Click()
 Screen.MousePointer = vbDefault
 Unload Me
End Sub

Private Sub Form_Load()
Me.Caption = SystemName & " " & Me.Caption
ClearFields
txtFields(0) = CurrentID
tblAccounts.Index = "UserAccount"
tblAccounts.Seek ">", CurrentID, 0
If tblAccounts.NoMatch = False Then
  If tblAccounts![User ID] = CurrentID Then LoadData
```

```
End If

End Sub

Private Sub Form_Unload(Cancel As Integer)
 Screen.MousePointer = vbDefault
End Sub
```

frmAccounts - 1

```
VERSION 5.00
Begin VB.Form frmAccounts
   BorderStyle     =   3  'Fixed Dialog
   Caption         =   "Account Setup"
   ClientHeight    =   4230
   ClientLeft      =   1095
   ClientTop       =   330
   ClientWidth     =   6255
   Icon            =   (Icon)
   LinkTopic       =   "Form2"
   LockControls    =   -1  'True
   MaxButton       =   0   'False
```

```
MinButton      =   0   'False
ScaleHeight     =   4230
ScaleWidth      =   6255
ShowInTaskbar   =   0   'False
StartUpPosition =   2   'CenterScreen
Begin VB.TextBox txtFields
   DataField       =   "Monthly Date"
   DataSource      =   "datPrimaryRS"
   Height          =   285
   Index           =   4
   Left            =   5400
   MaxLength       =   10
   TabIndex        =   35
   Top             =   3300
   Width           =   315
End
Begin VB.CheckBox chkDays
   Alignment       =   1  'Right Justify
   Caption         =   "Report on Monday:"
   DataField       =   "Monday"
   DataSource      =   "datPrimaryRS"
   Height          =   285
   Index           =   1
   Left         =  3660
   TabIndex     =  34
   Top          =  1740
   Width        =  2055
End
Begin VB.CheckBox chkDays
   Alignment    =  1  'Right Justify
   Caption      =  "Report on Tueday:"
   DataField    =  "Tuesday"
   DataSource   =  "datPrimaryRS"
   Height       =  285
   Index        =  2
   Left         =  3660
   TabIndex     =  33
   Top          =  2055
   Width        =  2055
End
Begin VB.CheckBox chkDays
   Alignment    =  1  'Right Justify
   Caption      =  "Report on Wednesday:"
   DataField    =  "Wednesday"
   DataSource   =  "datPrimaryRS"
   Height       =  285
```

```
      Index       =  3
      Left       =  3660
      TabIndex      =  32
      Top        =  2385
      Width       =  2055
     End


    frmAccounts - 2



     Begin VB.CheckBox chkDays
       Alignment     =  1  'Right Justify
       Caption      =  "Report on Thursday:"
       DataField     =  "Thursday"
       DataSource     =  "datPrimaryRS"
       Height      =  285
       Index       =  4
       Left       =  3660
       TabIndex      =  31
       Top        =  2700
       Width       =  2055
     End
     Begin VB.CheckBox chkDays
```

```
      Caption        =  "Report Monthly only on the        th"
      DataField      =  "Monthly Only"
      DataSource     =  "datPrimaryRS"
      Height         =  285
      Index          =  6
      Left           =  3180
      TabIndex       =  30
      Top            =  3315
      Width          =  2775
   End
   Begin VB.CheckBox chkDays
      Alignment      =  1  'Right Justify
      Caption        =  "Report on Friday:"
      DataField      =  "Friday"
      DataSource     =  "datPrimaryRS"
      Height         =  285
      Index          =  5
      Left           =  3660
      TabIndex       =  29
      Top            =  3000
      Width          =  2055
   End
   Begin VB.TextBox txtFields
```

```
  DataField      =  "Name"
  DataSource     =  "datPrimaryRS"
  Height       =  285
  Index       =  3
  Left       =  2040
  MaxLength      =  50
  TabIndex      =  2
  Top        =  1080
  Width       =  3435
End
Begin VB.TextBox txtFields
  DataField      =  "Reference Number"
  DataSource     =  "datPrimaryRS"
  Height       =  285
  Index       =  2
  Left       =  2040
  MaxLength      =  10
  TabIndex      =  1
  Top        =  720
  Width       =  1575
End
Begin VB.CheckBox chkFields
  DataField      =  "Extract Ascii"
```

```
    DataSource     =  "datPrimaryRS"
    Height       =  285
    Index        =  1
    Left        =  2040
    TabIndex      =  8
```

frmAccounts - 3

```
    Top          =  3000
    Width        =  315
  End
  Begin VB.CheckBox chkFields
    DataField     =  "Extract Qwicken"
    DataSource     =  "datPrimaryRS"
    Height       =  285
    Index        =  0
    Left        =  2040
    TabIndex      =  9
    Top          =  3315
    Width        =  315
  End
```

```
Begin VB.ComboBox Combo1
  DataField      =  "Account Type"
  DataSource     =  "datPrimaryRS"
  Height         =  315
  Left           =  2040
  Style          =  2  'Dropdown List
  TabIndex       =  3
  Top            =  1425
  Width          =  1575
End
Begin VB.PictureBox picButtons
  Appearance     =  0  'Flat
  BorderStyle    =  0  'None
  ForeColor      =  &H80000008&
  Height         =  360
  Left           =  0
  ScaleHeight    =  360
  ScaleWidth     =  6345
  TabIndex       =  18
  Top            =  3780
  Width          =  6345
  Begin VB.CommandButton cmdNExt
    Caption        =  "&Next"
```

```
   Height        =  300
   Left        =  4200
   TabIndex      =  28
   Top         =  0
   Width        =  975
 End
 Begin VB.CommandButton cmdPrevious
   Caption       =  "&Previous"
   Height        =  300
   Left        =  3180
   TabIndex      =  27
   Top         =  0
   Width        =  975
 End
 Begin VB.CommandButton cmdClose
   Caption       =  "&Close"
   Height        =  300
   Left        =  5220
   TabIndex      =  22
   Top         =  0
   Width        =  975
 End
 Begin VB.CommandButton cmdUpdate
```

```
      Caption       =   "&Update"
      Height        =   300
      Left          =   2145
      TabIndex      =   21
      Top           =   0

  frmAccounts - 4

      Width         =   975
    End
    Begin VB.CommandButton cmdDelete
      Caption       =   "&Delete"
      Height        =   300
      Left          =   1095
      TabIndex      =   20
      Top           =   0
      Width         =   975
    End
    Begin VB.CommandButton cmdAdd
      Caption       =   "&Add"
      Height        =   300
      Left          =   60
      TabIndex      =  19
      Top          =  0
      Width         =  975
    End
  End
  Begin VB.CheckBox chkFields
    DataField      =  "Attach Statement"
    DataSource      =  "datPrimaryRS"
    Height        =  285
    Index         =  6
    Left          =  2040
    TabIndex       =  7
    Top          =  2700
    Width         =  315
  End
  Begin VB.CheckBox chkFields
    DataField      =  "Report NSF"
    DataSource      =  "datPrimaryRS"
    Height        =  285
    Index         =  5
    Left          =  2040
    TabIndex       =  6
    Top          =  2385
```

```
      Width         =  315
   End
   Begin VB.CheckBox chkFields
      DataField      =  "Report Transactions"
      DataSource     =  "datPrimaryRS"
      Height         =  285
      Index          =  4
      Left           =  2040
      TabIndex       =  5
      Top            =  2055
      Width          =  315
   End
   Begin VB.CheckBox chkFields
      DataField      =  "Report Balance"
      DataSource     =  "datPrimaryRS"
      Height         =  285
      Index          =  3
      Left           =  2040
      TabIndex       =  4
      Top            =  1740
      Width          =  315
   End
   Begin VB.TextBox txtFields
```

```
DataField      =  "Account Number"

DataSource     =  "datPrimaryRS"

Height         =  285

Index          =  1
```

```
      Left          =  2040
      MaxLength      =  10
      TabIndex      =  0
      Top           =  380
      Width          =  1575
   End
   Begin VB.TextBox txtFields
      BackColor      =  &H80000000&
      DataField      =  "ID"
      DataSource      =  "datPrimaryRS"
      Height         =  285
      Index          =  0
      Left          =  2040
      Locked         =  -1  'True
      TabIndex       =  11
      TabStop        =  0   'False
      Top            =  60
      Width           =  735
   End
   Begin VB.Label lblLabels
```

```
    Alignment     =  1  'Right Justify
    Caption       =  "Last Report:"
    Height        =  255
    Index         =  11
    Left          =  4320
    TabIndex      =  36
    Top           =  60
    Width         =  1815
  End
  Begin VB.Label lblLabels
    Caption       =  "Account Name:"
    Height        =  255
    Index         =  10
    Left          =  120
    TabIndex      =  26
    Top           =  1080
    Width         =  1815
  End
  Begin VB.Label lblLabels
    Caption       =  "Reference Number:"
    Height        =  255
    Index         =  9
    Left          =  120
  TabIndex      =  25
  Top         =  720
  Width        =  1815
End
Begin VB.Label lblLabels
  Caption       =  "Extract ASCII:"
  Height       =  255
  Index        =  8
  Left        =  120
  TabIndex      =  24
  Top         =  3000
  Width        =  1815
End
Begin VB.Label lblLabels
  Caption       =  "Extract Qwicken:"
  Height       =  255
  Index        =  7
  Left        =  120
  TabIndex      =  23
  Top         =  3315
  Width        =  1815
```

```
End
Begin VB.Label lblLabels
  Caption        =  "Attach Statement:"
  Height         =  255
  Index          =  6
  Left           =  120
  TabIndex       =  17
  Top            =  2700
  Width          =  1815
End
Begin VB.Label lblLabels
  Caption        =  "Report NSF:"
  Height         =  255
  Index          =  5
  Left           =  120
  TabIndex       =  16
  Top            =  2385
  Width          =  1815
End
Begin VB.Label lblLabels
```

```
      Caption       =  "Report Transactions:"
      Height        =  255
      Index         =  4
      Left          =  120
      TabIndex      =  15
      Top           =  2055
      Width         =  1815
   End
   Begin VB.Label lblLabels
      Caption       =  "Report Balance:"
      Height        =  255
      Index         =  3
      Left          =  120
      TabIndex      =  14
      Top           =  1740
      Width         =  1815
   End
   Begin VB.Label lblLabels
      Caption       =  "Account Type:"
      Height        =  255
      Index         =  2
      Left          =  120
      TabIndex      =  13
   Top          =  1425
   Width        =  1815
 End
 Begin VB.Label lblLabels
   Caption      =  "Account Number:"
   Height       =  255
   Index        =  1
   Left         =  120
   TabIndex     =  12
   Top          =  380
   Width        =  1815
 End
 Begin VB.Label lblLabels
   Caption      =  "Customer ID:"
   Height       =  255
   Index        =  0
   Left         =  120
   TabIndex     =  10
   Top          =  60
   Width        =  1815
 End
```

End

frmAutotime - 1

```
Option Explicit
Private Sub cmdApply_Click()
SaveSetting App.Title, "Settings", "AutoTime", lblTime
SaveSetting App.Title, "Settings", "AutoBetween", lblBetween
cmdCancel_Click
End Sub
Private Sub cmdCancel_Click()
Unload Me
End Sub

Private Sub Form_Load()
Me.Caption = SystemName & " " & Me.Caption
lblTime = GetSetting(App.Title, "Settings", "AutoTime", "08:00 AM")
lblBetween = GetSetting(App.Title, "Settings", "AutoBetween", "15")
End Sub

Private Sub UpDown1_DownClick()
lblTime = Format$(DateAdd("n", -5, lblTime), "hh:mm AMPM")
End Sub
```

```
Private Sub UpDown1_UpClick()

lblTime = Format$(DateAdd("n", 5, lblTime), "hh:mm AMPM")

End Sub


Private Sub UpDown2_DownClick()

Dim L%

L% = Val(lblBetween)

If L% = 5 Then Exit Sub

lblBetween = L% - 5

DoEvents

End Sub


Private Sub UpDown2_UpClick()

Dim L%

L% = Val(lblBetween)

If L% = 60 Then Exit Sub

lblBetween = L% + 5

DoEvents

End Sub


frmAutotime - 1
```

```
VERSION 5.00
Object = "{86CF1D34-0C5F-11D2-A9FC-0000F8754DA1}#2.0#0"; "MSCOMCT2.OCX"
Begin VB.Form frmAutotime
  BorderStyle    =  3  'Fixed Dialog
  Caption        =  "Auto Setup"
  ClientHeight   =  1770
  ClientLeft     =  45
  ClientTop      =  330
  ClientWidth    =  4260
  LinkTopic      =  "Form1"
  MaxButton      =  0  'False
  MinButton      =  0  'False
  ScaleHeight    =  1770
  ScaleWidth     =  4260
  ShowInTaskbar  =  0  'False
  StartUpPosition =  1  'CenterOwner
  Begin VB.CommandButton cmdCancel
    Caption        =  "&Cancel"
    Height         =  300
    Left           =  2340
    TabIndex       =  7
    Top            =  1320
    Width          =  975
```

```
End
Begin VB.CommandButton cmdApply
  Caption        =  "&Save"
  Height         =  300
  Left           =  900
  TabIndex       =  6
  Top            =  1320
  Width          =  975
End
Begin MSComCtl2.UpDown UpDown1
  Height         =  285
  Left           =  3480
  TabIndex       =  1
  Top            =  420
  Width          =  240
  _ExtentX       =  423
  _ExtentY       =  503
  _Version       =  393216
  Enabled        =  -1  'True
End
Begin MSComCtl2.UpDown UpDown2
  Height         =  285
  Left           =  3480
```

```
    TabIndex      =  4
    Top        =  780
    Width       =  240
    _ExtentX      =  423
    _ExtentY      =  503
    _Version      =  393216
    Enabled       =  -1  'True
  End
  Begin VB.Label Label4
    Caption       =  "Minutes between Attempts:"
    Height        =  255
    Left        =  420
    TabIndex       =  5
    Top          =  780
    Width         =  2055
  End
  Begin VB.Label lblBetween
    Alignment      =  1  'Right Justify
    BackColor     =  &H8000000E&
    BorderStyle   =  1  'Fixed Single
    Caption       =  "15"
    Height        =  285
    Left          =  2580
    TabIndex      =  3
    Top           =  780
    Width         =  795
  End
  Begin VB.Label Label2
    Caption       =  "Begin e-Mail:"
    Height        =  255
    Left          =  420
    TabIndex      =  2
    Top           =  420
    Width         =  1335
  End
  Begin VB.Label lblTime
    Alignment     =  1  'Right Justify
    BackColor     =  &H8000000E&
    BorderStyle    =  1 'Fixed Single
    Caption        =  "08:00 AM"
    Height         =  285
    Left           =  2580
    TabIndex       =  0
    Top            =  420
    Width          =  795
  End
End
```

frmBroadcast - 1

```
Option Explicit

Private Sub cmdCancel_Click()
Unload Me
Set frmBroadcast = Nothing
End Sub

Private Sub cmdSend_Click()
If Combo1 = "" Then
  MsgBox "This e-mail has no send to address.", vbInformation, "NOTHING TO DO"
  Exit Sub
End If
If Trim$(Text2) = "" Then
  If MsgBox("This e-mail has no subject. Do you want to send anyway?", vbQuestion
+
vbYesNo, "NO SUBJECT") <> vbYes Then Exit Sub
End If

MAPISession1.SignOn
MAPIMessages1.SessionID = MAPISession1.SessionID
tblUsers.MoveFirst
While Not tblUsers.EOF
If ((Combo1.ListIndex = 2 Or (Combo1.ListIndex = 1 And tblUsers!Charge = True)) And
t
blUsers!Suspend = False) Or (Combo1.ListIndex > 1 And tblUsers![Sort Name] =
Combo1)
Then
   MAPIMessages1.Compose
   MAPIMessages1.RecipDisplayName = tblUsers![Address Name]
   MAPIMessages1.RecipAddress = tblUsers![e-Mail Address]
   MAPIMessages1.MsgSubject = Text2
   MAPIMessages1.MsgNoteText = Text1
   MAPIMessages1.Send
   If Combo1.ListIndex > 2 Then GoTo Endit
End If
tblUsers.MoveNext
Wend
Endit:
MAPISession1.SignOff
cmdCancel_Click
End Sub

Private Sub Form_Load()
Me.Caption = SystemName & " " & Me.Caption
Combo1.AddItem ""
Combo1.AddItem " All Users"
Combo1.AddItem " All Paying Users"
tblUsers.Index = "Sort Name"
tblUsers.MoveFirst
While Not tblUsers.EOF
  Combo1.AddItem tblUsers![Sort Name]
  tblUsers.MoveNext
Wend
Combo1.ListIndex = 0
End Sub
```

```
VERSION 5.00
Object = "{20C62CAE-15DA-101B-B9A8-444553540000}#1.1#0"; "MSMAPI32.OCX"
Begin VB.Form frmBroadcast
  BorderStyle   =  3  'Fixed Dialog
  Caption      =  "e-Mail Broadcast"
  ClientHeight   =  4890
  ClientLeft    =  45
  ClientTop     =  330
  ClientWidth    =  7980
  Icon        =  (Icon)
  LinkTopic     =  "Form1"
  LockControls   =  -1  'True
  MaxButton     =  0   'False
  MinButton     =  0   'False
  ScaleHeight    =  4890
  ScaleWidth    =  7980
  ShowInTaskbar   =  0   'False
  StartUpPosition =  2  'CenterScreen
  Begin MSMAPI.MAPIMessages MAPIMessages1
    Left        =  2000
    Top          =   2100
    _ExtentX     =   1005
    _ExtentY     =   1005
    _Version     =   393216
    AddressEditFieldCount=   1
    AddressModifiable=   0   'False
    AddressResolveUI=   0   'False
    FetchSorted     =   0   'False
    FetchUnreadOnly =   0   'False
 End
 Begin MSMAPI.MAPISession MAPISession1
    Left         =   3240
    Top          =   2100
    _ExtentX     =   1005
    _ExtentY     =   1005
    _Version     =   393216
    DownloadMail    =   0   'False
    LogonUI         =   -1  'True
    NewSession      =   -1  'True
    Password        =   "estatement"
    UserName        =   "estatement"
 End
 Begin VB.TextBox Text2
```

```
    Height      =  315
    Left       =  2040
    TabIndex      =  1
    Top         =  600
    Width       =  5715
  End
  Begin VB.CommandButton cmdCancel
    Cancel       =  -1  'True
    Caption       =  "&Cancel"
    Height       =  375
    Left        =  4080
    TabIndex      =  4
    Tag         =  "Cancel"
    Top         =  4320
    Width        =  1095
  End
  Begin VB.CommandButton cmdSend
    Caption       =  "&Send"
    Height       =  375
    Left        =  2580
    TabIndex      =  3
```

```
      Tag            =  "OK"
      Top            =  4320
      Width          =  1095
   End
   Begin VB.TextBox Text1
      Height         =  3015
      Left           =  180
      MultiLine      =  -1  'True
      ScrollBars     =  2  'Vertical
      TabIndex       =  2
      Top            =  1080
      Width          =  7575
   End
   Begin VB.ComboBox Combo1
      Height         =  315
      Left           =  2040
      Sorted         =  -1  'True
      Style          =  2  'Dropdown List
      TabIndex       =  0
      Top            =  180
      Width          =  3315
   End
```

```
Begin VB.Label Label1
   Caption       =  "Subject:"
   Height        =  255
   Index         =  1
   Left          =  900
   TabIndex      =  6
   Top           =  600
   Width         =  915
End
Begin VB.Label Label1
   Caption       =  "Send to:"
   Height        =  255
   Index         =  0
   Left          =  900
   TabIndex      =  5
   Top           =  240
   Width         =  915
End
End

frmGenerate - 1


Option Explicit
```

```
Dim FilNum%, CancelFlag%
Sub ReadLoans()  'Read the loan trial balance information
Dim A$, B$, C$, X%, FilNum%, Account$, FileName$, Pages%, CurrentPage%,
OldIndex$
OldIndex$ = tblAccounts.Index
tblAccounts.Index = "Account Number"
tblBalances.Index = "PrimaryKey"
For X% = 1 To 2
   If X% = 1 Then FileName$ = "Tbal005.prn" Else FileName$ = "Tbal007.prn"
   If Dir(ReportPath & FileName) = "" Then GoTo NextFile
   FilNum% = FreeFile
   Open ReportPath & FileName For Input As FilNum%
   Line Input #FilNum%, A$
   Line Input #FilNum%, A$
   LoanDate = Mid$(A$, 47, 2) & "-" & Mid$(A$, 49, 2) & "-" & Mid$(A$, 51, 2)
   LoanDate = Format$(LoanDate, "mm-dd-yyyy")
   Pages% = Val(Mid$(A$, 74, 4))
   CurrentPage% = 0
   Label1 = ""
   Label2 = "Reading " & FileName$ & "..."
   ProgressBar1.Value = 0
   ProgressBar1.Visible = True
   DoEvents
```

```
  While Not EOF(FilNum%)
   Line Input #FilNum%, A$
    A$ = RTrim$(A$)
    If A$ = Chr$(12) Then
      CurrentPage% = CurrentPage% + 1
          If CurrentPage% < Pages% Then ProgressBar1.Value = (CurrentPage% /
Pages%)
* 100
      DoEvents
      GoTo NextLine
    End If
    If Len(A$) < 80 Then GoTo NextLine
    Account$ = Trim$(Mid$(A$, 2, 11))
    If Val(Account$) > 0 Then            'New Loan
      tblAccounts.Seek "=", Account$
      If tblAccounts.NoMatch = True Then GoTo NextLine
      Line Input #FilNum%, B$
      Line Input #FilNum%, C$
      With tblBalances
        Label1 = "Adding Loan " & Account$
        DoEvents
        .Seek "=", Account$
        If .NoMatch Then .AddNew Else .Edit
```

```
            On Error Resume Next
            ![Account] = Account$
            ![Last Update] = LoanDate
            ![Balance] = CCur(Trim$(Mid$(A$, 51, 12)))
            If IsDate(Mid$(A$, 124, 8)) Then ![Date of Last Deposit] = Format$(Mid$
    (A$, 124, 8))
            If IsDate(Mid$(B$, 63, 8)) Then ![Maturity Date] = Format$(Mid$(B$, 63,
    8), "mm/dd/yyyy")
            If IsDate(Mid$(C$, 63, 8)) Then ![Next Due Date] = Format$(Mid$(C$, 63,
    8), "mm/dd/yyyy")
            ![Net Payoff] = CCur(Trim$(Mid$(C$, 50, 12)))
            ![Rate] = CSng(Trim$(Mid$(C$, 76, 7)))
            ![Payment] = CCur(Trim$(Mid$(A$, 74, 10)))
            ![YTD Interest] = CCur(Trim$(Mid$(C$, 85, 11)))
            ![Misc] = Trim$(Mid$(B$, 14, 33))
            ![Last Deposit] = CCur(Trim$(Mid$(B$, 73, 10))) 'Past Due Amount
            If Err = 0 Then .Update Else .CancelUpdate
            On Error GoTo 0
         End With

    frmGenerate - 2


       End If
```

```
NextLine:
  Wend
NextFile:
Close FilNum%
Next X%
ProgressBar1.Visible = False
Label2 = ""
Label1 = ""
tblAccounts.Index = OldIndex$
End Sub
Function FNum(A As String) As String
Dim L%
A$ = Trim$(A$)
L% = Len(A$)
If L% > 4 Then
  FNum = String(L% - 4, "x") & Right$(A$, 4)
Else
  FNum = A$
End If
End Function


Function Strip(AA$) As String
```

```
Do
 If Left$(AA$, 1) = Chr$(10) Then
   AA$ = Mid$(AA$, 2)
 Else
   Exit Do
 End If
Loop
Do
 If Right(AA$, 1) = Chr$(10) Then
   AA$ = Left$(AA$, Len(AA$) - 1)
 Else
   Exit Do
 End If
Loop
Strip = AA$
End Function
Sub MakeStatements()
Dim FilNum%, A$, S$(8), Account$, FilNum2%, X%, Page%, FileName$, D$, SDate$,
TCD$, O
IdAccount$, L%
Account$ = "": OldAccount$ = "":
Dim TestLine$
If ReportPath & "Stmt090.PRN" = "" Then Exit Sub
```

```
Label2 = "Reading account statements..."
tblAccounts.Index = "Account Number"
FilNum% = FreeFile
Open ReportPath & "STMT090.prn" For Input As FilNum%
Line Input #FilNum%, A$ ' Header Line
While Not EOF(FilNum%)
  Line Input #FilNum%, A$
  If A$ <> Chr$(12) Then GoTo NextLine
  'Read next 8 line into the buffer
NextSLine:
  Erase S$()
  For X% = 1 To 8
    Line Input #FilNum%, S$(X%)
    If S$(X%) = Chr$(12) Then GoTo NextSLine
  Next
  Account$ = Mid$(S$(5), 54, 8)
  Label1 = Account$
  DoEvents
```

frmGenerate - 3

```
  Page% = Val(Trim$(Mid$(S$(5), 78, 3)))
```

```
   SDate = Mid$(S$(6), 73, 8)
   tblAccounts.Seek "=", Val(Account$)
   If tblAccounts.NoMatch Then GoTo NextLine

   If tblAccounts![Attach Statement] = False Then GoTo NextLine
   Label1 = "Creating statement for " & Account$
   DoEvents
   FilNum2% = FreeFile
  FileName$ = Format$(Val(Account$), "00000000") & Format$(SDate$, "mm-dd-yy")
& ".
HTM"
   If Account$ <> OldAccount$ Then
     Open App.Path & "\" & FileName For Output As FilNum2%
     Print #FilNum2%, "<P><IMG SRC= " & GIFName & " <FONT COLOR=#FF0000>
<FONT SIZE
=" & Format$(StatementFont%, "##") & ">" & " " & SystemName & " from "; BankName
& "
<SUB><SMALL>Member FDIC</SMALL></SUB></FONT></P>"
     Print #FilNum2%, "<P><PRE><FONT SIZE=2><FONT COLOR=#0000FF>"
                                   Print #FilNum2%,
"=============================================================
================"
    Print #FilNum2%, "<B>NOTE:  For your convenience only! This statement does not
```

```
represent official"
    Print #FilNum2%, "      bank records.  Your statement will be mailed to you.
  Refer to your"
    Print #FilNum2%, "      mailed copy for important disclosures concerning your
account.</B>"
                                        Print #FilNum2%,
"===============================================================
================="
    Print #FilNum2%, "<FONT COLOR=#000000>"
  Else
    Open App.Path & "\" & FileName For Append As FilNum2%
    Print #FilNum2%, "<P><PRE><FONT SIZE=2><FONT COLOR=#FF0000>"
    Print #FilNum2%, String(36, 45) & " Page " & Page% & " " & String(36, 45)
    Print #FilNum2%, "<FONT COLOR=#000000>"
  End If
  OldAccount$ = Account$
  Print #FilNum2%, "* Your account name, address   *" & Mid$(Strip(S$(4)), 42)
     TestLine$ = "* and a portion of your account*" & Mid$(Strip(S$(5)), 42)
 L% = InStr(TestLine$, Account$): If L% <> 0 Then Mid$(TestLine$, L%, (Len(Account
$) - 4)) = String$(Len(Account$) - 4, "x")
  Print #FilNum2%, TestLine$
  Print #FilNum2%, "* number have been omitted for *" & Mid$(Strip(S$(6)), 42)
  Print #FilNum2%, "* your security.             *" & Mid$(Strip(S$(7)), 42)
```

```
    Print #FilNum2%, String(32, " ") & Mid$(Strip(S$(8)), 42)
    While Not EOF(FilNum%)
     Line Input #FilNum%, A$
     If A$ = Chr$(12) Then   'New Statement
       Print #FilNum2%, "<FONT></P>"
       Close FilNum2%
       GoTo NextSLine
     End If
     TestLine$ = Strip(A$)
    L% = InStr(TestLine$, Account$): If L% <> 0 Then Mid$(TestLine$, L%, (Len(Accou
not$) - 4)) = String(Len(Account$) - 4, "x")
     If Mid$(TestLine$, 41, 7) = "CREDITS" Then Mid$(TestLine$, 5, 4) = "xxxx"
    If Mid$(TestLine$, 29, 12) = "ACCOUNT NO. " Then Mid$(TestLine$, 43, 4) = "xxxx
"
     L% = InStr(TestLine$, "ACCT.--"): If L% <> 0 Then Mid$(TestLine$, L% + 7, 4) =
"xxxx"
     If InStr(A$, "NUMBER OF ITEMS") = 0 Then Print #FilNum2%, TestLine$
    Wend
    Print #FilNum2%, "<FONT></P>"
    Close FilNum2%
NextLine:
  Wend
```

```
 Close FilNum%
 Close FilNum2%
 tblAccounts.Index = "Account Seq"
End Sub

Private Sub cmdCancel_Click()
CancelFlag% = True
Screen.MousePointer = 0
Close FilNum%
Unload Me
Set frmGenerate = Nothing
End Sub

Private Sub Form_Load()
'Verify that necessary files are present
Me.Caption = SystemName & " " & Me.Caption
Dim HoldPath$
HoldPath$ = ReportPath
If ReportSet = "Prior" Then ReportPath = ReportPath & "LATEST\": ReportDateAdd =
-1
```

```
On Error GoTo 0

Show

DoEvents


Dim A$, AA$(2), SQL$, X%, f%, FileName$, L%, SCount%, FileNum3%, Pdate$,
HasAttachmen
t%, TCD$, SLoc&
Dim ECount&


tblAccounts.Index = "Account Seq"
tblTrans.Index = "Reference"
Screen.MousePointer = 11


'Build Statements
If GStatement% = True Then MakeStatements



Label1 = "Extracting Account Balances..."
DoEvents


'Delete old balance and Update New
If GBalance% <> True Then GoTo NextFunction1
SQL = "Delete * from Balances"
```

```
dbs.Execute SQL
SQL = "Update Accounts Set Sequence = 1"
dbs.Execute SQL
If GLoans% = True Then ReadLoans

If ReportPath & "MNTB002.PRN" <> "" Then
  FilNum% = FreeFile
  For f% = 2 To 4
    FileName$ = "MNTB" & Format$(f%, "000") & ".prn"
    Open ReportPath & FileName For Input As FilNum%
    Do Until EOF(FilNum%)
      Line Input #FilNum%, A$
      DoEvents: If CancelFlag = True Then Exit Sub
      'A$ = Trim$(A$)
      AA$(1) = Left$(A$, 64)
      AA$(2) = Mid$(A$, 68)
      For X% = 1 To 2
        A$ = Trim$(Left$(AA$(X), 10))
        If Val(A$) > 0 Then
          tblAccounts.Seek "=", A$, 1
          If tblAccounts.NoMatch = False Then
            Label1 = "Adding Balance for " & A$
            DoEvents
            tblBalances.AddNew
             tblBalances!Account = Val(A$)
             Select Case f%
                Case 2, 3
                   tblBalances!Balance = CCur(Trim$(Mid$(AA$(X%), 25, 17)))
                   If IsDate(Trim$(Right$(AA$(X), 9))) Then tblBalances![Date
of Last Deposit] = Trim$(Right$(AA$(X%), 9))
                   tblBalances![Last Deposit] = CCur(Trim$(Mid$(AA$(X%), 39, 1
7)))
                Case 4
                   tblBalances!Balance = CCur(Trim$(Mid$(AA$(X%), 25, 13)))
                   If IsDate(Mid$(AA$(X), 47, 8)) Then tblBalances![Date of La
st Deposit] = Mid$(AA$(X%), 47, 8)
                   If IsDate(Mid$(AA$(X), 38, 8)) Then tblBalances![Maturity D
ate] = Mid$(AA$(X%), 38, 8)
                   tblBalances![Last Deposit] = CCur(Trim$(Mid$(AA$(X%), 55, 1
07)))
                Case Else
             End Select
            tblBalances.Update
```

```
          End If
        End If
      Next
    Loop
    Close FilNum%
Next 'f%
End If

NextFunction1:
If GTrans <> True Then GoTo NextFunction2
'Load New Transactions
tblAccounts.Index = "REFERENCE"
If ReportPath & "PJNL001.PRN" <> "" Then
  FilNum% = FreeFile
  FileName$ = "PJNL001.PRN"
  Open ReportPath & FileName For Input As FilNum%
  Line Input #FilNum%, A$
  Line Input #FilNum%, A$
  Pdate$ = Mid$(A$, 68, 2) & "-" & Mid$(A$, 70, 2) & "-" & Mid$(A$, 72, 2)
  Pdate$ = Format$(Pdate$, "mm-dd-yyyy")
  tblTrans.Index = "Duplicate"
  Do Until EOF(FilNum%)
    Line Input #FilNum%, A$
```

```
DoEvents: If CancelFlag = True Then Exit Sub
If Len(A$) < 100 Then GoTo NextLine
AA$(1) = Left$(A$, 64)
AA$(2) = Mid$(A$, 71)
For X% = 1 To 2
  A$ = Trim$(Left$(AA$(X), 9)) & Mid$(AA$(X), 11, 2)
'    If A$ = "116160" Then Stop
  If Val(A$) > 100000 Then
    tblAccounts.Seek "=", A$
    If tblAccounts.NoMatch = False Then
      Label1 = "Adding Transaction(s) for " & A$
      DoEvents
      tblTrans.Seek "=", Val(A$), Pdate$, Val(Mid$(AA$(X%), 44, 6)) + 0, CCur
(Trim$(Mid$(AA$(X%), 19, 13)))
      If tblTrans.NoMatch = True Then
        tblTrans.AddNew
          tblTrans!Reference = Val(A$)
          tblTrans!TC = Mid$(AA$(X), 15, 3)
          tblTrans!Amount = CCur(Trim$(Mid$(AA$(X%), 19, 13)))
          tblTrans!Source = Mid$(AA$(X%), 34, 4)
          tblTrans!Serial = Val(Mid$(AA$(X%), 44, 6)) + 0
```

```
              tblTrans!Date = Pdate$
             tblTrans.Update
            End If
           End If
          End If
      Next
   NextLine:
    Loop
    Close FilNum%
    tblAccounts.Index = "Account Seq"
    tblTrans.Index = "Reference"
   End If

   'Delete NSF and Add new
   SQL = "Delete * from NSF"
   dbs.Execute SQL
   If ReportPath & "CF__026.PRN" <> "" Then
     Dim Acc$, Bal As Currency, B$, SAcc$
     FilNum% = FreeFile
     FileName$ = "CF__026.PRN"
     Open ReportPath & FileName For Input As FilNum%
     Do Until EOF(FilNum%)
    Line Input #FilNum%, A$
    DoEvents: If CancelFlag = True Then Exit Sub
    If Left$(A$, 20) = "--------------------" Then Acc$ = "": Bal = 0: GoTo NextNSF
    Acc$ = Trim$(Mid$(A$, 5, 8))
    If Acc$ = "" Then GoTo NextNSF
    If Val(Acc$) = 0 Then GoTo NextNSF
    'If Acc$ = "124990" Then Stop
    tblAccounts.Seek "=", Acc$, 1
    If tblAccounts.NoMatch = False Then
      tblAccounts.Edit
      tblAccounts!Sequence = 0
      tblAccounts.Update
      B$ = Mid$(A$, 106, 15)
      'User Reverinst here
      L% = InStrRev(B$, "*")
      B$ = Mid$(B$, L% + 1)
      SAcc$ = Acc$
      Bal = CCur(B$)
      Label1 = "Adding NSF for " & Acc$
      DoEvents
    End If
NextNSF:
    If Bal <> 0 And Mid$(A$, 115, 1) = "." Then
```

```
        tblNSF.AddNew
        tblNSF!Account = SAcc$
        tblNSF!Balance = Bal
        tblNSF!Amount = CCur(Mid$(A$, 106, 12))
        tblNSF!Serial = Val(Mid$(A$, 80, 6))
        tblNSF.Update
      End If
  Loop
End If
Close FilNum%


NextFunction2:
'Build e-mail
Dim T$, D$
MAPISession1.SignOn
MAPIMessages1.SessionID = MAPISession1.SessionID
tblUsers.MoveFirst
tblAccounts.Index = "User Seq"
tblBalances.Index = "PrimaryKey"

frmGenerate - 7
```

```
tblNSF.Index = "Account"
While Not tblUsers.EOF

 SCount% = 0
 DoEvents: If CancelFlag = True Then Exit Sub

 'Check for suspened user'
 If tblUsers!Suspend = True Then GoTo NextUser2

 'Build the first part of the email message

 MAPIMessages1.Compose
 MAPIMessages1.RecipDisplayName = tblUsers![Address Name]
 MAPIMessages1.RecipAddress = tblUsers![e-Mail Address]
 MAPIMessages1.MsgSubject = SystemName & " for " & Format$(Pdate$, "dddd mmm
dd, yy
yy")

 'Has the user confirmed usage? if not send a confirmation
 If tblUsers!Confirmed = False Then
  MAPIMessages1.MsgSubject = "eStatement address confirmation " & Format$(Now,
"m
m/dd/yyyy")
```

```
    T$ = "Dear Customer: " & Chr$(10) & Chr$(10)
    T$ = T$ & "Please confirm receipt of this eStatement Address Confirmation by" &
Chr$(10)
    T$ = T$ & "replying to this e-mail.  Click the REPLY button of your" & Chr$(10)
    T$ = T$ & "e-mail program and then the SEND button.  No other action is" & Chr$
(10)
    T$ = T$ & "required. Upon receipt of your confirmation we will begin" & Chr$(10
)
    T$ = T$ & "daily eStatement services." & Chr$(10) & Chr$(10)
    T$ = T$ & "Thank you for your business..." & Chr$(10) & Chr$(10)
    T$ = T$ & "   The Lamar Bank eStatement Team"
    T$ = T$ & "   eStatement@lamarbanktexas.com "
    GoTo NextUser
  End If

  'Build Text for email
  T$ = SystemName$ & " from " & BankName$ & "   Member FDIC" & Chr$(10)
  T$ = T$ & "As of close of business " & Format$(Pdate$, "dddd mmm-dd-yyyy") &
Chr$(
10)
  tblAccounts.Seek ">=", tblUsers(0), 0
  If tblAccounts![User ID] <> tblUsers![ID] Then GoTo NextUser
  While Not tblAccounts.EOF
    If tblAccounts![User ID] <> tblUsers![ID] Then GoTo NextUser
    tblNSF.Seek "=", tblAccounts![Account Number]
    If tblNSF.NoMatch = False Then
                        If T$ <> "" Then T$ = T$ &
"==================================================
=" & Chr$(10)
      T$ = T$ & "------------NSF NOTICE--NSF NOTICE-------------" & Chr$(10)
      T$ = T$ & "NSF Account:   " & FNum(tblNSF!Account) & " - " & tblAccounts!Nam
e & Chr$(10)
      T$ = T$ & "       Balance:   " & Format$(tblNSF!Balance, "Currency") & Chr
$(10) & Chr$(10)
      T$ = T$ & "NSF Checks:" & Chr$(10)
      T$ = T$ & "            SERIAL    AMOUNT" & Chr$(10)
      T$ = T$ & "            ---------- ----------------" & Chr$(10)
      While Not tblNSF.EOF
        If tblNSF!Account <> tblAccounts![Account Number] Then GoTo NextNSF2
        T$ = T$ & "            " & Format$(tblNSF!Serial, "000000    ") & Righ
t$(Space$(12) & Format$(tblNSF!Amount, "Currency"), 12) & Chr$(10)
        tblNSF.MoveNext
      Wend
    End If
NextNSF2:
   'Check here for report day
   If tblAccounts![Monthly Only] = False And tblAccounts(Weekday(Pdate$) + 10) = F
alse Then GoTo NextAccount3
    'code to check date for Monthly only here
    tblBalances.Seek "=", tblAccounts![Account Number]
    If tblBalances.NoMatch = False And GBalance = True Then
                                If T$ <> "" Then T$ = T$ &
"==================================================
====" & Chr$(10)
        D$ = Dir$(App.Path & "\" & Format$(tblAccounts![Account Number], "0000000
0") & "*.HTM")
        If D$ <> "" And GStatement% = True Then
          T$ = T$ & "A Statement for this account is attached today!" & Chr$(10)
        End If
      Select Case tblAccounts![Account Type]
        Case "Checking"
          T$ = T$ & "  Checking Account:   " & FNum(tblAccounts![Account Number
]) & " - " & tblAccounts!Name & Chr$(10)
```

```
              If tblBalances!Balance = -99999999.99 Then
                T$ = T$ & "There is a full balance hold on this account!" & Chr$(1
0)
              Else
                T$ = T$ & "  Available Balance:    " & Format$(tblBalances!Balance
, "Currency") & Chr$(10)
              End If
              If tblBalances![Last Deposit] <> 0 Then T$ = T$ & "        Last Depo
sit:   " & Format$(tblBalances![Last Deposit], "Currency") & " on " & Format$(tblBal
ances![Date of Last Deposit], "mm/dd/yyyy") & Chr$(10)
              T$ = T$ & Chr$(10)
            Case "Savings"
              T$ = T$ & "   Savings Account:    " & FNum(tblAccounts![Account Numbe
r]) & " - " & tblAccounts!Name & Chr$(10)
              If tblBalances!Balance = -99999999.99 Then
                T$ = T$ & "There is a full balance hold on this account!" & Chr$(1
0)
              Else
                T$ = T$ & "  Available Balance:    " & Format$(tblBalances!Balance
, "Currency") & Chr$(10)
              End If
              If tblBalances![Last Deposit] <> 0 Then T$ = T$ & "        Last Depo
sit:   " & Format$(tblBalances![Last Deposit], "Currency") & " on " & Format$(tblBal
ances![Date of Last Deposit], "mm/dd/yyyy") & Chr$(10)
      T$ = T$ & Chr$(10)
    Case "CD"
      T$ = T$ & "  Certificate Number:   " & FNum(tblAccounts![Account Numb
er]) & " - " & tblAccounts!Name & Chr$(10)
      If tblBalances!Balance = -99999999.99 Then
        T$ = T$ & "There is a full balance hold on this account!" & Chr$(1
0)
      Else
        T$ = T$ & "   Available Balance:    " & Format$(tblBalances!Balanc
e, "Currency") & Chr$(10)
      End If
      T$ = T$ & "        Maturity Date:    " & Format$(tblBalances![Maturi
ty Date], "mm/dd/yyyy") & Chr$(10)
      T$ = T$ & "  Next Interest Date:    " & Format$(tblBalances![Date of
Last Deposit], "mm/dd/yyyy") & Chr$(10)
      If tblBalances![Last Deposit] <> 0 Then T$ = T$ & "     Accrued Inter
est:      " & Format$(tblBalances![Last Deposit], "Currency") & Chr$(10)
      T$ = T$ & Chr$(10)
    Case "Loan"
      If Format$(LoanDate, "Short Date") <> Format$(Pdate$, "Short Date") T
hen
```

```
        T$ = T$ & "  All Information as of:   " & Format$(LoanDate, "Short Date") & Chr$(10)
      End If
      T$ = T$ & "           Loan Number:  " & FNum(tblAccounts![Account Number]) & " - " & tblAccounts!Name & Chr$(10)
      T$ = T$ & "        Current Balance:  " & Format$(tblBalances![Balance], "Currency") & Chr$(10)
      If tblBalances![Last Deposit] > 0 Then T$ = T$ & "     Past Due Amount:    " & Format$(tblBalances![Last Deposit], "Currency") & Chr$(10)
      T$ = T$ & "            Interest Rate:   " & Format$(tblBalances![Rate] / 100, "###.00 %") & Chr$(10)
      T$ = T$ & "       Payment Amount:  " & Format$(tblBalances![Payment], "Currency") & Chr$(10)
      T$ = T$ & "     Last Payment Date:  " & Format$(tblBalances![Date of Last Deposit], "Short Date") & Chr$(10)
      T$ = T$ & "          Next Due Date:  " & Format$(tblBalances![Next Due Date], "Short Date") & Chr$(10)
      T$ = T$ & "            Maturity Date:  " & Format$(tblBalances![Maturity Date], "Short Date") & Chr$(10)
      T$ = T$ & " Payoff (call to verify):  " & Format$(tblBalances![Net Pa
yoff], "Currency") & Chr$(10)
          T$ = T$ & "  Interest Year-to-date:  " & Format$(tblBalances![YTD Int
erest], "Currency") & Chr$(10)
          T$ = T$ & "Collateral Description:  " & tblBalances![Misc] & Chr$(10)
          T$ = T$ & Chr$(10)
        Case Else
          T$ = ""
      End Select
    End If
    'Add Trans
    If GTrans% = True Then
      tblTrans.Seek ">=", tblAccounts![Reference Number], 0
      If tblTrans.NoMatch Then GoTo NextAccount
    If tblTrans!Reference <> tblAccounts![Reference Number] Then GoTo NextAccount
      T$ = T$ & "Transaction(s) for Account " & FNum(tblAccounts![Account Number])
& Chr$(10)
      T$ = T$ & "Date         Serial #        Amount            Transaction
Source and Type" & Chr$(10)
      T$ = T$ & "---------  ----------   -------------- -------------------
-------------" & Chr$(10)
      While Not tblTrans.EOF
        If tblTrans!Reference = tblAccounts![Reference Number] Then
          tblTranCodes.Seek "=", tblTrans!TC
          If tblTranCodes.NoMatch Then
            MsgBox "An unknown trancode has been found: " & tblTrans!TC, vbInforma
tion, "UNKNOWN TRANCODE": TCD$ = "Unknow"
           Else
            TCD$ = tblTranCodes(1)
            If tblTrans!Source <> "MICR" Then TCD$ = tblTrans!Source & "   " & TCD
$
          End If
          T$ = T$ & Format$(tblTrans!Date, "mm/dd/yy") & "     " & Format$(tblTrans
!Serial, "000000  ") & Right$(Space$(17) & Format$(tblTrans!Amount, "Currency"), 17)
 & Space$(8) & TCD$ & Chr$(10)
        Else
          GoTo LastTran  'End of Trans
        End If
        tblTrans.MoveNext
      Wend
    End If
LastTran:
    If T$ <> "" Then T$ = T$ & Chr$(10)

NextAccount2:
    'If flaged, build CSV File for trans
    If GCSV = False Or (GCSV = True And tblAccounts![Extract ASCII] = False) Then
GoTo NextAccount
    tblTrans.Seek "=", tblAccounts![Account Number]
    If tblTrans.NoMatch Then GoTo NextAccount
    Dim FilNum3%, TranType$
    FilNum3% = FreeFile
   Open App.Path & "\" & Format$(tblAccounts![Account Number], "00000000") & ".CS
V" For Output As #FilNum3%
    While Not tblTrans.EOF
     If tblTrans!Reference = tblAccounts![Reference Number] Then
      tblTranCodes.Seek "=", tblTrans!TC
      If tblTranCodes.NoMatch = False Then
        If tblTranCodes!Action = -1 Then
          Write #FilNum3%, tblAccounts![Account Number], Pdate$, "Debit", tblTr
ans!Serial, Format$(-tblTrans!Amount, "#########.##")
        Else
          Write #FilNum3%, tblAccounts![Account Number], Pdate$, "Credit", tblT
rans!Serial, Format$(tblTrans!Amount, "#########.##")
        End If
      End If
     Else
```

```
        Close FilNum3%
        GoTo NextAccount
      End If
      tblTrans.MoveNext
     Wend
     Close FilNum3%
NextAccount:
    D$ = Dir$(App.Path & "\" & Format$(tblAccounts![Account Number], "00000000") &
"
.CSV")
    If D$ <> "" And GCSV = True Then
     HasAttachment = True
     MAPIMessages1.AttachmentIndex = SCount%
     MAPIMessages1.AttachmentPathName = App.Path & "\" & D$
       MAPIMessages1.AttachmentName = Pdate$ & " Transactions for Account " &
FNum(tbl
Accounts![Account Number]) & ".CSV"
     SCount% = SCount% + 1
    End If
    D$ = Dir$(App.Path & "\" & Format$(tblAccounts![Account Number], "00000000") &
"
*.HTM")
    If D$ <> "" And GStatement% = True Then
```

```
      HasAttachment = True
      MAPIMessages1.AttachmentIndex = SCount%
      MAPIMessages1.AttachmentPathName = App.Path & "\" & D$
      MAPIMessages1.AttachmentName = Format$(Mid$(D$, 9, 8), "mm-dd-yyyy") & "
Statem
ent for Account " & FNum(tblAccounts![Account Number]) & ".HTM"
      SCount% = SCount% + 1
    End If
    tblAccounts.Edit
    tblAccounts![Last Report] = Now
    tblAccounts.Update
NextAccount3:
    tblAccounts.MoveNext
Wend        'End of account loop

NextUser:
 If T$ <> "" Or HasAttachment = True Then
   Label1 = "eMailing " & MAPIMessages1.RecipDisplayName
   DoEvents
   If T$ <> "" Then
      If GMessage <> "" Then T$ = GMessage & Chr$(10) & T$
      MAPIMessages1.MsgNoteText = Left$(T$, Len(T$) - 1)
   Else
```

```
    T$ = GMessage & Chr$(10)
    MAPIMessages1.MsgNoteText = "Statement attached"
  End If
  MAPIMessages1.Send
  ECount = ECount + 1
  DoEvents
 End If
NextUser2:
 T$ = ""
 HasAttachment = False
 tblUsers.MoveNext
Wend     'End of User Loop

'End Of Processing
Screen.MousePointer = 0
On Error Resume Next
MAPISession1.SignOff

If ECount > 1 Then
   tblHistory.AddNew
```

```
      tblHistory![Generation Date] = Now
      tblHistory!Statements = ECount
    tblHistory.Update
    tblOptions.Edit
      tblOptions![Last Generated] = Now
    tblOptions.Update
End If


'Remove reported transactions
tblAccounts.MoveFirst
While Not tblAccounts.EOF
  If Format$(tblAccounts![Last Report], "mm/dd/yyyy") = Format$(Now, "mm/dd/yyyy")
T
hen
   SQL = "Delete * from Transaction where Reference = '" & Trim(tblAccounts![Referen
ce Number]) & "'"
    dbs.Execute SQL
  End If
  tblAccounts.MoveNext
Wend
ReportPath = HoldPath
ReportDateAdd = 0
Kill App.Path & "\*.HTM"
Kill App.Path & "\*.CSV"
On Error GoTo 0
cmdCancel_Click
End Sub
```

frmGenerate - 1

```
VERSION 5.00
Object = "{831FDD16-0C5C-11D2-A9FC-0000F8754DA1}#2.0#0"; "MSCOMCTL.OCX"
Object = "{20C62CAE-15DA-101B-B9A8-444553540000}#1.1#0"; "MSMAPI32.OCX"
Begin VB.Form frmGenerate
   BorderStyle     =   3  'Fixed Dialog
   Caption         =   "Generate e-Mail"
   ClientHeight    =   1650
   ClientLeft      =   45
   ClientTop       =   330
   ClientWidth     =   4680
   Icon            =   (Icon)
   LinkTopic       =   "Form1"
   MaxButton       =   0   'False
   MinButton       =   0   'False
   ScaleHeight     =   1650
```

```
ScaleWidth      =   4680
ShowInTaskbar   =   0   'False
StartUpPosition =   1   'CenterOwner
Begin MSComctlLib.ProgressBar ProgressBar1
   Height          =   255
   Left            =   300
   TabIndex        =   3
   Top             =   780
   Visible         =   0   'False
   Width           =   4035
   _ExtentX        =   7117
   _ExtentY        =   450
   _Version        =   393216
   Appearance      =   1
End
Begin VB.CommandButton cmdCancel
   Cancel          =   -1  'True
   Caption         =   "&Cancel"
   Height          =   300
   Left            =   1740
   TabIndex        =   1
   Top             =   1200
   Width           =   1095
```

```
End
Begin MSMAPI.MAPIMessages MAPIMessages1
  Left          =  3840
  Top            =  960
  _ExtentX       =  1005
  _ExtentY       =  1005
  _Version       =  393216
  AddressEditFieldCount=  1
  AddressModifiable=  0  'False
  AddressResolveUI=  0  'False
  FetchSorted     =  0  'False
  FetchUnreadOnly =  0  'False
End
Begin MSMAPI.MAPISession MAPISession1
  Left          =  3960
  Top            =  60
  _ExtentX       =  1005
  _ExtentY       =  1005
  _Version       =  393216
  DownloadMail    =  0  'False
  LogonUI        =  -1  'True
  NewSession      =  -1  'True
  Password        =  "ebalance"
  UserName     =  "ebalance"
 End
 Begin VB.Label Label2

frmGenerate - 2

    Alignment     =  2  'Center
    AutoSize      =  -1  'True
    Height        =  195
    Left          =  2280
    TabIndex      =  2
    Top           =  60
    Width         =  75
  End
  Begin VB.Label Label1
    Alignment     =  2  'Center
    AutoSize      =  -1  'True
    Height        =  195
    Left          =  2280
    TabIndex      =  0
    Top           =  420
    Width         =  75
  End
```

```
End
```

frmLogin - 1

```
Option Explicit

Private Declare Function GetUserName Lib "advapi32.dll" Alias "GetUserNameA"
(ByVal l
pbuffer As String, nSize As Long) As Long

Public OK As Boolean
Private Sub Form_Load()
  Dim sBuffer As String
  Dim lSize As Long
  Me.Caption = SystemName & " Login"
  sBuffer = Space$(255)
  lSize = Len(sBuffer)
  Call GetUserName(sBuffer, lSize)
  If lSize > 0 Then
    txtUserName.Text = Left$(sBuffer, lSize)
  Else
    txtUserName.Text = vbNullString
  End If
End Sub

Private Sub cmdCancel_Click()
  OK = False
  Me.Hide
End Sub

Private Sub cmdOK_Click()
  'ToDo: create test for correct password
  'check for correct password
  If txtPassword.Text = "" Then
    OK = True
    Me.Hide
  Else
    MsgBox "Invalid Password, try again!", , "Login"
    txtPassword.SetFocus
    txtPassword.SelStart = 0
    txtPassword.SelLength = Len(txtPassword.Text)
  End If
End Sub
```

```
VERSION 5.00
Begin VB.Form frmLogin
  BorderStyle    =  3  'Fixed Dialog
  Caption        =  "Login"
  ClientHeight   =  1590
  ClientLeft     =  45
  ClientTop      =  330
  ClientWidth    =  3750
  Icon           =  (Icon)
  LinkTopic      =  "Form1"
  MaxButton      =  0  'False
  MinButton      =  0  'False
  ScaleHeight    =  1590
  ScaleWidth     =  3750
  ShowInTaskbar  =  0  'False
  StartUpPosition =  2  'CenterScreen
  Tag            =  "Login"
  Begin VB.CommandButton cmdCancel
    Cancel         =  -1  'True
    Caption        =  "Cancel"
```

```
      Height          =   360
      Left            =   2100
      TabIndex        =   5
      Tag             =   "Cancel"
      Top             =   1020
      Width           =   1140
   End
   Begin VB.CommandButton cmdOK
      Caption         =   "OK"
      Default         =   -1  'True
      Height          =   360
      Left            =   495
      TabIndex        =   4
      Tag             =   "OK"
      Top             =   1020
      Width           =   1140
   End
   Begin VB.TextBox txtPassword
      Height          =   285
      IMEMode         =   3  'DISABLE
      Left            =   1305
      PasswordChar    =   "*"
      TabIndex        =   1
```

```
  Top          =  525
  Width         =  2325
End
Begin VB.TextBox txtUserName
  Height        =  285
  Left         =  1305
  TabIndex       =  3
  Top          =  135
  Width         =  2325
End
Begin VB.Label lblLabels
  Caption        =  "&Password:"
  Height        =  248
  Index         =  1
  Left         =  105
  TabIndex       =  0
  Tag          =  "&Password:"
  Top          =  540
  Width         =  1080
End
Begin VB.Label lblLabels
frmLogin - 2
```

```
        Caption       =  "&User Name:"
        Height       =  248
        Index        =  0
        Left         =  105
        TabIndex      =  2
        Tag          =  "&User Name:"
        Top          =  150
        Width         =  1080
      End
   End
   frmMain - 1



   Option Explicit

   Private Declare Function OSWinHelp% Lib "user32" Alias "WinHelpA" (ByVal hwnd&,
   ByVal
    HelpFile$, ByVal wCommand%, dwData As Any)
   Sub OpenFiles()
     FilePath = GetSetting(App.Title, "Settings", "FilePath", App.Path)
     Set wrkJet = CreateWorkspace("", "admin", "", dbUseJet)
      ' Open Database object from saved Microsoft Jet database
```

```
 Set dbs = wrkJet.OpenDatabase(FilePath & "\eStatement.mdb", False)


 Set tblUsers = dbs.OpenRecordset("Users")
 Set tblAccounts = dbs.OpenRecordset("Accounts")
 Set tblBalances = dbs.OpenRecordset("Balances")
 Set tblTrans = dbs.OpenRecordset("Transaction")
 Set tblNSF = dbs.OpenRecordset("NSF")
 Set tblTranCodes = dbs.OpenRecordset("Tran Codes")
 Set tblHistory = dbs.OpenRecordset("History")
 Set tblOptions = dbs.OpenRecordset("Options")
 tblOptions.Index = "Primary"
 tblOptions.MoveFirst
 tblHistory.Index = "Primary"
 tblTranCodes.Index = "Primary"
End Sub
 Sub CloseFiles()
 On Error Resume Next
 tblUsers.Close
 tblAccounts.Close
 tblBalances.Close
 tblTrans.Close
 tblNSF.Close
 tblTranCodes.Close
 dbs.Close

 On Error GoTo 0

 End Sub


Private Sub Form_Load()


BackPath:  'Get the path to the reports

 ReportPath = GetSetting(App.Title, "Settings", "ReportPath", "")

 LoanPath = GetSetting(App.Title, "Settings", "LoanPath", "")


 'Check the Paths

 If ReportPath = "" Then

   If MsgBox("Please select the path to the report files.", vbOKCancel, "SETUP") =
vbCancel Then mnuExit_Click

   mnuChangeReportPath_Click

   GoTo BackPath

 End If


 If LoanPath = "" Then

   If MsgBox("Please select the path to the loan data files.", vbOKCancel, "SETUP")
 = vbCancel Then mnuExit_Click

   mnuChangeLoanPath_Click

   GoTo BackPath
```

```
End If


'Open the Data File

OpenFiles

'Center the Autoemail form

fraAuto.Left = (Me.Width - fraAuto.Width) / 2

fraAuto.Top = (Me.Height - fraAuto.Height) / 3

'Check to see if AutoEMail was on at the last shutdown

  If InStr(UCase$(Command), "AUTO") > 0 Or GetSetting(App.Title, "Settings",
"AutoEma
il", "OFF") = "ON" Then


frmMain - 2



  'Start if it was on

  mnuEnableAutoeMail_Click

 End If


'Get System Settings from Registery


 Me.Caption = SystemName & " - " & BankName & " - Ver: " & App.Major & "." &
App.Min
```

```
or & App.Revision
 Me.Show
 DoEvents
 If BankName = "Demo Bank" Then mnuNameChange_Click
 If GetSetting(App.Title, "Settings", "Users", "") = "Disabled" Then
  UsersDisabled = True
  mnuEnable_Click
 End If

End Sub

Private Sub Form_Resize()
 fraAuto.Left = (Me.Width - fraAuto.Width) / 2
 fraAuto.Top = (Me.Height - fraAuto.Height) / 3
 Line1.X1 = 0
 Line1.X2 = Me.Width
End Sub

Private Sub Form_Unload(Cancel As Integer)
 Dim i As Integer
 CloseFiles
 'close all sub forms
 For i = Forms.Count - 1 To 1 Step -1
```

```
   Unload Forms(i)
  Next


End Sub


Private Sub mnuBroadcast_Click()
frmBroadcast.Show vbModal, Me
End Sub


Private Sub mnuChangeLoanPath_Click()
 dlgCommonDialog.ShowOpen
  If dlgCommonDialog.FileName <> "" Then
    Dim L%
    L% = InStrRev(dlgCommonDialog.FileName, "\")
    If L% > 0 Then
      LoanPath = Left(dlgCommonDialog.FileName, L%)
    Else
      LoanPath = dlgCommonDialog.FileName
    End If
    If ReportPath <> "" Then SaveSetting App.Title, "Settings", "LoanPath", LoanPath
  End If
 End Sub
```

```
Private Sub mnuChangeReportPath_Click()
 dlgCommonDialog.ShowOpen
 If dlgCommonDialog.FileName <> "" Then
   Dim L%
   L% = InStrRev(dlgCommonDialog.FileName, "\")
   If L% > 0 Then
     ReportPath = Left(dlgCommonDialog.FileName, L%)


frmMain - 3



   Else
     ReportPath = dlgCommonDialog.FileName
   End If
   If ReportPath <> "" Then SaveSetting App.Title, "Settings", "ReportPath", Report
Path
  End If
End Sub


Private Sub mnuDataUsers_Click()
 Dim f As New frmUsersGrid
 f.Show vbModal, Me
End Sub
```

```
Private Sub mnuDisable_Click()
If UsersDisabled = True Then Exit Sub
Dim SQL$
SQL$ = "Update Users Set [Previous State] = Suspend"
dbs.Execute SQL
SQL$ = "Update Users Set Suspend = True"
dbs.Execute SQL
MsgBox "All users disabled!", vbInformation
tblUsers.Index = "PrimaryKey"
tblUsers.Seek "=", 10
tblUsers.Edit
tblUsers!Suspend = False
tblUsers.Update
UsersDisabled = True
lblUsersDisabled.Visible = True
SaveSetting App.Title, "Settings", "Users", "Disabled"
End Sub


Private Sub mnuDisableAutoeMail_Click()
If MsgBox("WARNING! Your are about to disable auto email." & vbCrLf & "Daily
statemen
ts will not be automatically  generated." & vbCrLf & vbCrLf & "Is this what you want
to do?", vbExclamation + vbDefaultButton2 + vbYesNo, "WARNING!") <> vbYes Then
Exit S
ub
fraAuto.Visible = False
Timer1.Enabled = False
mnuEnableAutoeMail.Visible = True
mnuDisableAutoeMail.Visible = False
AutoFlag = False
SaveSetting App.Title, "Settings", "AutoEmail", "OFF"
DoEvents
End Sub


Private Sub mnuEditAutoEmail_Click()
 Dim f As New frmAutotime
 f.Show vbModal, Me
End Sub


Private Sub mnuEnable_Click()
If UsersDisabled = False Then Exit Sub
Dim SQL$
SQL$ = "Update Users Set Suspend = [Previous State]"
dbs.Execute SQL
UsersDisabled = False
```

```
lblUsersDisabled.Visible = False
SaveSetting App.Title, "Settings", "Users", "Enabled"
End Sub

Private Sub mnuEnableAutoeMail_Click()
Dim StartTime As Date
If UsersDisabled = True Then mnuEnable_Click
SaveSetting App.Title, "Settings", "AutoEmail", "ON"

frmMain - 4

StartTime = GetSetting(App.Title, "Settings", "AutoTime", "08:00 AM")
LastComplete = GetSetting(App.Title, "Settings", "LastAuto", "08:00 AM")

If DateValue(LastComplete) = DateValue(Now()) Then
  NextTry = Format$(DateAdd("w", 1, LastComplete), "mm/dd/yyyy ") & StartTime
Else
  NextTry = Format$(DateAdd("w", 1, LastComplete), "mm/dd/yyyy ") & StartTime
End If
'Add a day if it is Sunday
If Weekday(NextTry) = 7 Then NextTry = DateAdd("w", 1, NextTry)
'Add another day if it Monday
```

```
If Weekday(NextTry) = 1 Then NextTry = DateAdd("w", 1, NextTry)

AutoI = GetSetting(App.Title, "Settings", "AutoBetween", "15")
lblLastComplete = LastComplete
lblLastTry = "Waiting"
lblNext = NextTry
lblMissing = 0
fraAuto.Visible = True
mnuEnableAutoeMail.Visible = False
mnuDisableAutoeMail.Visible = True
lblTime = Format$(Now, "hh:mm:ss AMPM")
Timer1.Enabled = True
AutoFlag = True
DoEvents
End Sub

Private Sub mnuExit_Click()
Unload Me
End
End Sub

Private Sub mnuGenerateCurrent_Click()
ReportSet = "Current"
```

```
If GetSetting(App.Title, "Settings", "FilesVerified", "") <> Now Then
 mnuVerify_Click
 If Format$(GetSetting(App.Title, "Settings", "FilesVerified", ""), "mm/dd/yyyy") <>
Format$(Now, "mm/dd/yyyy") Then
   If MsgBox("There are missing files needed to generate eMail.  Do you wish to con
tinue?", vbQuestion + vbYesNo, "INCOMPLETE INFORMATION") <> vbYes Then Exit
Sub
 End If
End If
On Error Resume Next
 GMessage = "": GBalance = True: GTrans% = True: GStatement% = True: GLoans%
= True:
 GCSV% = True: GQwick% = True
 frmGenerate.Show vbModal, Me
 SaveSetting App.Title, "Settings", "LastAuto", Now
On Error GoTo 0
End Sub


Private Sub mnuGeneratePrior_Click()
ReportSet = "Prior"
On Error Resume Next
 GMessage = "": GBalance = True: GTrans% = True: GStatement% = True: GLoans%
= True:
 GCSV% = True: GQwick% = True
  frmGenerate.Show vbModal, Me
On Error GoTo 0
End Sub

Private Sub mnuHelpAbout_Click()
  frmAbout.Show vbModal, Me
End Sub


Private Sub mnuHelpSearchForHelpOn_Click()
frmMain - 5



  Dim nRet As Integer

  'if there is no helpfile for this project display a message to the user
  'you can set the HelpFile for your application in the
  'Project Properties dialog
  If Len(App.HelpFile) = 0 Then
    MsgBox "Unable to display Help Contents. There is no Help associated with this pr
oject.", vbInformation, Me.Caption
  Else
    On Error Resume Next
    nRet = OSWinHelp(Me.hwnd, App.HelpFile, 261, 0)
    If Err Then
     MsgBox Err.Description
    End If
  End If

End Sub



Private Sub mnuHelpContents_Click()
  Dim nRet As Integer


  'if there is no helpfile for this project display a message to the user
  'you can set the HelpFile for your application in the
  'Project Properties dialog
  If Len(App.HelpFile) = 0 Then
    MsgBox "Unable to display Help Contents. There is no Help associated with this pr
oject.", vbInformation, Me.Caption
  Else
    On Error Resume Next
    nRet = OSWinHelp(Me.hwnd, App.HelpFile, 3, 0)
    If Err Then
     MsgBox Err.Description
    End If
  End If
End Sub


Private Sub mnuNameChange_Click()
frmNameChange.Show vbModal, Me
Me.Caption = SystemName & " - " & BankName & " - Ver: " & App.Major & "." &
App.Minor
 & App.Revision
End Sub


Private Sub mnuSelected_Click()
frmSelected.Show vbModal, Me
End Sub


Private Sub mnuTranCodes_Click()
frmTranCodes.Show vbModal, Me
End Sub


Private Sub mnuVerify_Click()
Dim FileName$
FilesMissing = 0
FileName = "MNTB002.PRN"
```

```
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "MNTB003.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "MNTB004.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "MNTB005.PRN"

frmMain - 6


If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "TBAL005.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "TBAL007.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "TBAL008.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
```

```
 = False Then GoSub ShowMissing
FileName = "PJNL001.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "PJNL002.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "STMT090.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "CFLZ007.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "CF__026.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
FileName = "CF__027.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1: If AutoFlag
 = False Then GoSub ShowMissing
If FilesMissing = 0 Then
   SaveSetting App.Title, "Settings", "FilesVerified", Now
   MsgBox "All files needed to generate email are present.", vbInformation, "VERIFIED
"
```

```
Else
  If AutoFlag = False Then
    If FilesMissing = 1 Then
      MsgBox "One file needed to generate eStatements is missing!", vbInformation,
"MISSING FILES"
    Else
      MsgBox FilesMissing & " files needed to generate eStatements are missing!",
vbInformation, "MISSING FILES"
    End If
  End If
End If
Exit Sub

'Show which file is missing if not Auto Generating
ShowMissing:
  MsgBox FileName & " is missing.", vbInformation, "MISSING FILE"
  Return
End Sub

Private Sub Timer1_Timer()
lblTime = Format$(Now, "hh:mm AMPM")
If Not AutoFlag Then Exit Sub
If Now < NextTry Then Exit Sub
```

```
'Give it a try
'Check Files
Dim X%, FileName$

FilesMissing = 0
FileName = "MNTB002.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "MNTB003.PRN"

frmMain - 7

If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "MNTB004.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "MNTB005.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "TBAL005.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "TBAL007.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "TBAL008.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "PJNL001.PRN"
```

```
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "PJNL002.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "STMT090.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "CFLZ007.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "CF__026.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1
FileName = "CF__027.PRN"
If Dir$(ReportPath & FileName) = "" Then FilesMissing = FilesMissing + 1


If FilesMissing = 0 Then
  SaveSetting App.Title, "Settings", "FilesVerified", Now
  DoEvents
Else
  lblMissing = FilesMissing
  lblLastTry = Format$(Now, "mm/dd/yyyy hh:mm:ss AMPM")
  LastTry = lblLastTry
  NextTry = DateAdd("n", AutoI, LastTry)
  lblNext = NextTry
  DoEvents
  Exit Sub
```

```
End If
On Error Resume Next
GMessage = "": GBalance = True: GTrans% = True: GStatement% = True: GLoans%
= True:
GCSV% = True: GQwick% = True
frmGenerate.Show vbModal, Me
SaveSetting App.Title, "Settings", "LastAuto", Now
mnuExit_Click
Exit Sub
LastComplete = Now
lblLastComplete = Format$(Now, "mm/dd/yyyy hh:mm:ss AMPM")
Dim StartTime As Date
StartTime = GetSetting(App.Title, "Settings", "AutoTime", "07:00 AM")
NextTry = Format$(DateAdd("w", 1, LastComplete), "mm/dd/yyyy ") & StartTime
FilesMissing = 0
lblMissing = 0
lblNext = NextTry
DoEvents


End Sub
frmMain - 1
```

```
VERSION 5.00
Object = "{F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.2#0"; "COMDLG32.OCX"
Begin VB.Form frmMain
  Caption       =  "eStatement Main Menu"
  ClientHeight   =  5640
  ClientLeft     =  165
  ClientTop      =  450
  ClientWidth    =  8415
  Icon          =  (Icon)
  LinkTopic      =  "Form1"
  ScaleHeight    =  5640
  ScaleWidth     =  8415
  StartUpPosition =  2  'CenterScreen
  Begin VB.Timer Timer1
    Interval      =  65535
    Left         =  840
    Top          =  1140
  End
  Begin VB.Frame fraAuto
    Caption       =  "Auto eMail is Active"
    ForeColor     =  &H000000FF&
    Height        =  2355
    Left         =  2340
```

```
TabIndex      =  1
Top           =  300
Visible       =  0  'False
Width         =  4035
Begin VB.Label lblTime
   BackColor     =  &H8000000E&
   BorderStyle   =  1  'Fixed Single
   Height        =  255
   Left          =  1560
   TabIndex      =  11
   Top           =  1860
   Width         =  1215
End
Begin VB.Label Label7
   Caption       =  "Current Time:"
   Height        =  255
   Left          =  300
   TabIndex      =  10
   Top           =  1860
   Width         =  1275
End
Begin VB.Label lblMissing
   BackColor     =  &H8000000E&
   BorderStyle    =  1  'Fixed Single
   Height       =  255
   Left        =  1560
   TabIndex      =  9
   Top         =  1500
   Width        =  615
End
Begin VB.Label Label6
   Caption       =  "Missing Files:"
   Height       =  255
   Left        =  300
   TabIndex      =  8
   Top         =  1500
   Width        =  1275
End
Begin VB.Label lblLastTry
   BackColor     =  &H8000000E&
   BorderStyle    =  1  'Fixed Single
```

frmMain - 2

```
   Height       =  255
```

```
    Left         =  1560
    TabIndex       =  7
    Top          =  1140
    Width         =  2295
  End
  Begin VB.Label Label5
    Caption        =  "Last Attempt:"
    Height        =  255
    Left         =  300
    TabIndex       =  6
    Top          =  1180
    Width         =  1275
  End
  Begin VB.Label lblLastComplete
    BackColor      =  &H8000000E&
    BorderStyle     =  1  'Fixed Single
    Height        =  255
    Left         =  1560
    TabIndex       =  5
    Top          =  420
    Width         =  2295
  End
  Begin VB.Label lblNext
```

```
    BackColor     =  &H8000000E&
    BorderStyle    =  1 'Fixed Single
    Height       =  255
    Left        =  1560
    TabIndex      =  4
    Top         =  780
    Width        =  2295
  End
  Begin VB.Label Label3
    Caption       =  "Next Schedule:"
    Height       =  255
    Left        =  300
    TabIndex      =  3
    Top         =  780
    Width        =  1275
  End
  Begin VB.Label Label2
    Caption       =  "Last Complete:"
    Height       =  255
    Left        =  300
    TabIndex      =  2
    Top         =  420
    Width        =  1275
  End
End
Begin MSComDlg.CommonDialog dlgCommonDialog
  Left         =  3180
  Top          =  1350
  _ExtentX     =  847
  _ExtentY     =  847
  _Version     =  393216
End
Begin VB.Label lblUsersDisabled
  Caption      =  "WARNING! USERS ARE DISABLED!"
  BeginProperty Font
    Name          =  "MS Sans Serif"
    Size          =  12
    Charset       =  0
    Weight        =  700
    Underline     =  0  'False
```

```
      Italic        =  0  'False
      Strikethrough  =  0  'False
    EndProperty
    ForeColor     =  &H000000FF&
    Height        =  255
    Left          =  240
    TabIndex      =  12
    Top           =  60
    Visible       =  0  'False
    Width         =  4635
  End
  Begin VB.Line Line1
    X1            =  0
    X2            =  11000
    Y1            =  0
    Y2            =  0
  End
  Begin VB.Label Label1
    Alignment     =  2  'Center
    AutoSize      =  -1  'True
    Height       =  195
    Left        =  4920
    TabIndex      =  0
    Top         =  1440
    Visible       =  0  'False
    Width        =  75
  End
  Begin VB.Menu mnuDataUsers
    Caption       =  "&Customers"
  End
  Begin VB.Menu mnuOptions
    Caption       =  "&Options"
    Begin VB.Menu mnuChangeReportPath
      Caption       =  "&Change Report Path"
    End
    Begin VB.Menu mnuChangeLoanPath
      Caption       =  "Change &Loan Data Path"
    End
    Begin VB.Menu mnuNameChange
      Caption       =  "Change &System Information"
    End
    Begin VB.Menu mnuEditAutoEmail
      Caption       =  "Edit Auto e-Mail &Settings"
    End
    Begin VB.Menu mnuTranCodes
      Caption       =  "&Edit Tran Codes"
    End
    Begin VB.Menu mnuEnable_Disable
      Caption       =  "&Enable/Disable"
      Begin VB.Menu mnuEnable
        Caption       =  "&Enable All"
      End
      Begin VB.Menu mnuDisable
        Caption       =  "&Disable All"
      End
    End
    Begin VB.Menu mnuSelected
      Caption       =  "&Generate Selected e-Mail"
    End
  End
  Begin VB.Menu mnuVerify
    Caption       =  "&Verify Files"
  End
  Begin VB.Menu mnuEnableAutoeMail
frmMain - 4
```

```
    Caption         =   "&Enable Auto e-Mail"
  End
  Begin VB.Menu mnuDisableAutoeMail
    Caption         =   "&Disable Auto e-Mail"
    Visible         =   0   'False
  End
  Begin VB.Menu mnuGenerate
    Caption         =   "&Generate e-Mail"
    Begin VB.Menu mnuGenerateCurrent
      Caption         =   "Generate &Current"
    End
    Begin VB.Menu mnuGeneratePrior
      Caption         =   "General &Prior"
    End
  End
  Begin VB.Menu mnuBroadcast
    Caption         =   "&Broadcast"
  End
  Begin VB.Menu mnuHelp
    Caption         =   "&Help"
    Begin VB.Menu mnuHelpContents
      Caption         =   "&Contents"
    End
    Begin VB.Menu mnuHelpSearchForHelpOn
      Caption        =  "&Search For Help On..."
    End
    Begin VB.Menu mnuHelpBar0
      Caption        =  "-"
    End
    Begin VB.Menu mnuHelpAbout
      Caption        =  "&About "
    End
  End
  Begin VB.Menu mnuExit
    Caption        =  "&Exit"
  End
End
```

frmNameChange - 1

```
Option Explicit

Private Sub cmdClose_Click()
Unload Me
Set frmNameChange = Nothing
End Sub

Private Sub cmdSave_Click()
Dim X%
X% = Val(txtStatementFont)
If X% > 8 Or X% < 2 Then
  MsgBox "Valid font sizes are from 2 to 8 only!" & vbCrLf & vbCrLf & "Try Again!",
vbInformation, "INVALID FONT SIZE"
  Exit Sub
End If
SaveSetting App.Title, "Settings", "SystemName", Trim$(txtSystemName)
SaveSetting App.Title, "Settings", "BankName", Trim$(txtBankName)
SaveSetting App.Title, "Settings", "StatementFont", Trim$(txtStatementFont)
SaveSetting App.Title, "Settings", "GIFName", Trim$(txtGifName)
SystemName$ = Trim$(txtSystemName)
```

```
BankName$ = Trim$(txtBankName)
StatementFont% = Trim$(txtStatementFont)
GIFName$ = Trim$(txtGifName)
cmdClose_Click
End Sub


Private Sub cmdBrowse_Click()
Dim FileName$, L%
FileName$ = GIFName
L% = InStrRev(FileName$, "\")
If L% <> 0 Then
  CommonDialog1.InitDir = Left$(FileName$, L%)
Else
  CommonDialog1.InitDir = App.Path
End If
CommonDialog1.Filter = "Pictures (*.gif;*.jpg)|*.gif;*.jpg"
CommonDialog1.FilterIndex = 1
CommonDialog1.CancelError = True
CommonDialog1.Action = 1
If Err > 0 Then On Error GoTo 0: Exit Sub
On Error GoTo 0
FileName$ = CommonDialog1.FileName
```

```
If UCase$(Right$(FileName$, 3)) <> "GIF" And UCase$(Right$(FileName$, 3)) <> "JPG"
Th
en
  MsgBox "Invalid file type.  JPG or GIF only.", vbInformation, "FILE TYPE ERROR"
  Exit Sub
End If
txtGifName = CommonDialog1.FileName


End Sub


Private Sub Form_Load()
  Me.Caption = SystemName & " " & Me.Caption
  txtSystemName = GetSetting(App.Title, "Settings", "SystemName", "eStatement")
  txtBankName = GetSetting(App.Title, "Settings", "BankName", "Demo Bank")
  txtStatementFont = GetSetting(App.Title, "Settings", "StatementFont", "6")
  txtGifName = GetSetting(App.Title, "Settings", "GIFName", "")
End Sub
```

```
VERSION 5.00
Object = "{F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.2#0"; "COMDLG32.OCX"
Begin VB.Form frmNameChange
   Caption         =   "Change System Information"
   ClientHeight    =   2535
   ClientLeft      =   60
   ClientTop       =   345
   ClientWidth     =   5430
   LinkTopic       =   "Form1"
   ScaleHeight     =   2535
   ScaleWidth      =   5430
   StartUpPosition =   1  'CenterOwner
   Begin MSComDlg.CommonDialog CommonDialog1
      Left            =   4260
      Top             =   2100
      _ExtentX        =   847
      _ExtentY        =   847
      _Version        =   393216
   End
   Begin VB.CommandButton cmdBrowse
```

```
  Caption     =  "&Browse"
  Height      =  315
  Left        =  4560
  TabIndex    =  11
  Tag         =  "OK"
  Top         =  1500
  Width       =  795
End
Begin VB.TextBox txtGifName
  Height      =  315
  Left        =  1740
  TabIndex    =  9
  Top         =  1500
  Width       =  2715
End
Begin VB.CommandButton cmdSave
  Caption     =  "&Save"
  Height      =  375
  Left        =  1418
  TabIndex    =  7
  Tag         =  "OK"
  Top         =  2025
  Width       =  1095
```

```
End
Begin VB.CommandButton cmdClose
  Cancel        =  -1  'True
  Caption       =  "&Close"
  Height        =  375
  Left          =  2918
  TabIndex      =  6
  Tag           =  "Cancel"
  Top           =  2025
  Width         =  1095
End
Begin VB.TextBox txtStatementFont
  Height        =  315
  Left          =  1740
  MaxLength     =  2
  TabIndex      =  5
  Top           =  1080
  Width         =  435
End
Begin VB.TextBox txtBankName
  Height        =  315
```

```
    Left        =  1740
    TabIndex      =  3
    Top         =  660
    Width        =  2715
  End
  Begin VB.TextBox txtSystemName
    Height       =  315
    Left        =  1740
    TabIndex      =  1
    Top         =  240
    Width        =  2715
  End
  Begin VB.Label Label1
    Caption       =  "Statement Graphic:"
    Height       =  255
    Index        =  4
    Left        =  120
    TabIndex      =  10
    Top         =  1500
    Width        =  1575
  End
  Begin VB.Label Label1
    Caption       =  "(2 to 8)"
    Height      =  255
    Index       =  3
    Left        =  2340
    TabIndex      =  8
    Top         =  1080
    Width       =  1575
  End
  Begin VB.Label Label1
    Caption      =  "Statement Font Size:"
    Height      =  255
    Index       =  2
    Left        =  120
    TabIndex      =  4
    Top         =  1080
    Width       =  1575
  End
  Begin VB.Label Label1
    Caption      =  "Bank Name:"
    Height      =  255
    Index       =  1
    Left        =  120
    TabIndex      =  2
    Top         =  660
    Width        =  1575
  End
  Begin VB.Label Label1
     Caption      =  "System Name:"
     Height       =  255
     Index        =  0
     Left       =  120
     TabIndex       =  0
     Top          =  240
     Width         =  1575
  End
End
frmSelected - 1


Option Explicit


Private Sub cmdCancel_Click()
Unload Me
Set frmSelected = Nothing
End Sub


Private Sub cmdContinue_Click()
```

```
Dim X%, T%
If Check1(0) = 1 Then GBalance% = True Else GBalance% = False
If Check1(1) = 1 Then GTrans% = True Else GTrans% = False
If Check1(2) = 1 Then GLoans% = True Else GLoans% = False
If Check1(3) = 1 Then GStatement% = True Else GStatement% = False
If Check1(4) = 1 Then GCSV% = True Else GCSV% = False
If Check1(5) = 1 Then GQwick% = True Else GQwick% = False
For X% = 0 To 5
   T% = T% + Check1(X%)
Next
If T% = 0 Then
  MsgBox "You must select at least one option to generate.", vbInformation, "NOTHING
 TO DO"
   Exit Sub
End If
GMessage = Trim$(Text1)
If Trim$(Text1) = "" Then
  If MsgBox("No message has been entered, do you wish to continue?", vbQuestion +
vb
YesNo + vbDefaultButton2, "MESSAGE OPTION") <> vbYes Then Exit Sub
End If
On Error Resume Next
Me.Hide
```

```
frmGenerate.Show
On Error GoTo 0
cmdCancel_Click
End Sub

Private Sub Form_Load()
Me.Caption = SystemName & " " & Me.Caption
End Sub
frmSelected - 1



VERSION 5.00
Begin VB.Form frmSelected
   BorderStyle     =   3  'Fixed Dialog
   Caption         =   "Selective Send"
   ClientHeight    =   4005
   ClientLeft      =   45
   ClientTop       =   330
   ClientWidth     =   4680
   Icon            =   (Icon)
   LinkTopic       =   "Form1"
   MaxButton       =   0   'False
   MinButton       =   0   'False
```

```
ScaleHeight    =  4005
ScaleWidth     =  4680
ShowInTaskbar  =  0  'False
StartUpPosition =  2  'CenterScreen
Begin VB.TextBox Text1
  Height       =  915
  Left         =  360
  MultiLine    =  -1  'True
  ScrollBars   =  2  'Vertical
  TabIndex     =  8
  Top          =  2580
  Width        =  4035
End
Begin VB.CommandButton cmdContinue
  Cancel       =  -1  'True
  Caption      =  "&Continue"
  Height       =  300
  Left         =  840
  TabIndex     =  7
  Top          =  3600
  Width        =  1095
End
Begin VB.CommandButton cmdCancel
```

```
      Caption        =  "C&ancel"
      Height         =  300
      Left           =  2580
      TabIndex       =  6
      Top            =  3600
      Width          =  1095
   End
   Begin VB.CheckBox Check1
      Caption        =  "Generate Qwicken Attachments"
      Height         =  255
      Index          =  5
      Left           =  960
      TabIndex       =  5
      Top            =  1980
      Width          =  2895
   End
   Begin VB.CheckBox Check1
      Caption        =  "Generate CSV Attachments"
      Height         =  255
      Index          =  4
      Left           =  960
      TabIndex       =  4
      Top            =  1620
   Width         =  2895
 End
 Begin VB.CheckBox Check1
   Caption       =  "Generate Statement Attachments"
   Height        =  255
   Index         =  3
frmSelected - 2



   Left          =  960
   TabIndex      =  3
   Top           =  1260
   Width         =  2895
 End
 Begin VB.CheckBox Check1
   Caption       =  "Generate Loan Information"
   Height        =  255
   Index         =  2
   Left          =  960
   TabIndex      =  2
   Top           =  900
   Width         =  2895
 End
```

```
Begin VB.CheckBox Check1
  Caption      =  "Generate Transactions"
  Height       =  255
  Index        =  1
  Left         =  960
  TabIndex     =  1
  Top          =  540
  Width        =  2895
End
Begin VB.CheckBox Check1
  Caption      =  "Generate Balance Information"
  Height       =  255
  Index        =  0
  Left         =  960
  TabIndex     =  0
  Top          =  180
  Width        =  2895
End
Begin VB.Label Label1
  Caption      =  "Message:"
  Height       =  255
  Left         =  60
  TabIndex     =  9
```

```
      Top         =  2280
      Width        =  1155
    End
  End
  frmSplash - 1



  Option Explicit


  Private Sub Form_Load()
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
    lblProductName.Caption = App.Title
    lblCompanyProduct = App.CompanyName
  '  lblCopyright = App.LegalCopyright
  '  lblCompany = App.CompanyName



  End Sub
  frmSplash - 1



  VERSION 5.00
  Begin VB.Form frmSplash
```

```
BorderStyle    =  3  'Fixed Dialog
ClientHeight   =  4710
ClientLeft     =  45
ClientTop      =  45
ClientWidth    =  7455
ControlBox     =  0  'False
LinkTopic      =  "Form1"
MaxButton      =  0  'False
MinButton      =  0  'False
ScaleHeight    =  4710
ScaleWidth     =  7455
ShowInTaskbar  =  0  'False
StartUpPosition =  2  'CenterScreen
Visible        =  0  'False
Begin VB.Frame fraMainFrame
   Height        =  4590
   Left          =  60
   TabIndex      =  0
   Top           =  -15
   Width         =  7380
   Begin VB.PictureBox picIcon
      AutoSize      =  -1  'True
      BackColor     =  &H00C0C0C0&
```

```
  ClipControls   =  0  'False
  Height        =  1260
  Left         =  420
  Picture        =  (Bitmap)
  ScaleHeight    =  1200
  ScaleMode      =  0  'User
  ScaleWidth     =  1200
  TabIndex       =  7
  TabStop        =  0  'False
  Top           =  600
  Width          =  1260
End
Begin VB.Label lblLicenseTo
  Alignment      =  1  'Right Justify
  Caption        =  "LicenseTo: Lamar Bank - Beaumont Texas "
  Height         =  255
  Left          =  270
  TabIndex       =  1
  Tag           =  "LicenseTo"
  Top           =  300
  Width          =  6855
End
Begin VB.Label lblProductName
```

```
      AutoSize      =  -1  'True
      Caption       =  "Product"
      BeginProperty Font
        Name          =  "Lucida Calligraphy"
        Size          =  24
        Charset       =  0
        Weight        =  700
        Underline     =  0   'False
        Italic        =  0   'False
        Strikethrough  =  0   'False
      EndProperty
      Height        =  615
      Left          =  2670
      TabIndex      =  6
      Tag           =  "Product"
      Top           =  1200
frmSplash - 2

      Width         =  3120
   End
   Begin VB.Label lblCompanyProduct
      AutoSize      =  -1  'True
```

```
  Caption       = "CompanyProduct"
  BeginProperty Font
    Name        = "MS Sans Serif"
    Size        = 13.5
    Charset     = 0
    Weight      = 700
    Underline     = 0 'False
    Italic      = 0 'False
    Strikethrough = 0 'False
  EndProperty
  Height      = 360
  Left        = 2505
  TabIndex      = 5
  Tag         = "CompanyProduct"
  Top         = 765
  Width       = 2415
End
Begin VB.Label lblPlatform
  Alignment     = 1 'Right Justify
  AutoSize      = -1 'True
  Caption       = "Platform Windows 95+"
  BeginProperty Font
    Name        = "MS Sans Serif"
```

```
      Size          =  13.5
      Charset        =  0
      Weight         =  700
      Underline       =  0   'False
      Italic         =  0   'False
      Strikethrough   =  0   'False
    EndProperty
    Height          =  360
    Left           =  3900
    TabIndex         =  4
    Tag            =  "Platform"
    Top            =  2400
    Width           =  3105
  End
  Begin VB.Label lblVersion
    Alignment       =  1  'Right Justify
    AutoSize        =  -1  'True
    Caption         =  "Version"
    BeginProperty Font
      Name           =  "MS Sans Serif"
      Size          =  12
      Charset        =  0
      Weight         =  700
      Underline     =  0  'False
      Italic        =  0  'False
      Strikethrough  =  0  'False
    EndProperty
    Height        =  300
    Left          =  6075
    TabIndex       =  3
    Tag           =  "Version"
    Top           =  2760
    Width          =  930
  End
  Begin VB.Label lblWarning
    Caption       =  "Warning:  Copyright 1999-2000, Resource Development Sys
    BeginProperty Font
```

frmSplash - 3

```
      Name          =  "MS Sans Serif"
      Size          =  9.75
      Charset        =  0
      Weight         =  700
      Underline      =  0  'False
```

```
        Italic        =  0  'False
        Strikethrough  =  0  'False
      EndProperty
      Height        =  615
      Left        =  300
      TabIndex      =  2
      Tag         =  "Warning"
      Top         =  3720
      Width        =  6855
    End
  End
End


frmTranCodes - 1



Option Explicit
Private Sub cmdAdd_Click()
  datPrimaryRS.Recordset.AddNew
End Sub


Private Sub cmdDelete_Click()
  With datPrimaryRS.Recordset
```

```
    .Delete

    .MoveNext

    If .EOF Then .MoveLast

  End With

End Sub


Private Sub cmdRefresh_Click()

  'This is only needed for multi user apps

  datPrimaryRS.Refresh

End Sub


Private Sub cmdUpdate_Click()

  datPrimaryRS.UpdateRecord

  datPrimaryRS.Recordset.Bookmark = datPrimaryRS.Recordset.LastModified

End Sub


Private Sub cmdClose_Click()

  Screen.MousePointer = vbDefault

  Unload Me

End Sub


Private Sub datPrimaryRS_Error(DataErr As Integer, Response As Integer)

  'This is where you would put error handling code
  'If you want to ignore errors, comment out the next line
  'If you want to trap them, add code here to handle them
  MsgBox "Data error event hit err:" & Error$(DataErr)
  Response = 0  'Throw away the error
End Sub


Private Sub datPrimaryRS_Reposition()
  Screen.MousePointer = vbDefault
  On Error Resume Next
  'This will display the current record position for dynasets and snapshots
  datPrimaryRS.Caption = "Trancode: " & (datPrimaryRS.Recordset.AbsolutePosition
+ 1)
End Sub


Private Sub datPrimaryRS_Validate(Action As Integer, Save As Integer)
  'This is where you put validation code
  'This event gets called when the following actions occur
  Select Case Action
    Case vbDataActionMoveFirst
    Case vbDataActionMovePrevious
    Case vbDataActionMoveNext
    Case vbDataActionMoveLast
    Case vbDataActionAddNew
    Case vbDataActionUpdate
    Case vbDataActionDelete
    Case vbDataActionFind
    Case vbDataActionBookmark
    Case vbDataActionClose
      Screen.MousePointer = vbDefault
  End Select
  Screen.MousePointer = vbHourglass
End Sub

Private Sub Form_Load()
Me.Caption = SystemName & " " & Me.Caption
frmTranCodes - 2


datPrimaryRS.DatabaseName = App.Path & "\estatement.mdb"
End Sub

Private Sub Form_Unload(Cancel As Integer)
  Screen.MousePointer = vbDefault
End Sub
```

```
VERSION 5.00
Begin VB.Form frmTranCodes
  BorderStyle    =  3  'Fixed Dialog
  Caption       =  "Tran Codes"
  ClientHeight   =  1695
  ClientLeft    =  1095
  ClientTop     =  330
  ClientWidth    =  5550
  Icon          =  (Icon)
  LinkTopic      =  "Form2"
  MaxButton      =  0  'False
  MinButton      =  0  'False
  ScaleHeight     =  1695
  ScaleWidth     =  5550
  ShowInTaskbar  =  0  'False
  StartUpPosition =  2  'CenterScreen
  Begin VB.PictureBox picButtons
    Align         =  2  'Align Bottom
    Appearance      =  0  'Flat
    BorderStyle     =  0  'None
```

```
ForeColor     =  &H80000008&
Height        =  300
Left          =  0
ScaleHeight   =  300
ScaleWidth    =  5550
TabIndex      =  6
Top           =  1050
Width         =  5550
Begin VB.CommandButton cmdClose
  Caption       =  "&Close"
  Height        =  300
  Left          =  4505
  TabIndex      =  11
  Top           =  0
  Width         =  975
End
Begin VB.CommandButton cmdUpdate
  Caption       =  "&Update"
  Height        =  300
  Left          =  3409
  TabIndex      =  10
  Top           =  0
  Width         =  975
```

```
End
Begin VB.CommandButton cmdRefresh
  Caption      =  "&Refresh"
  Height       =  300
  Left         =  2313
  TabIndex     =  9
  Top          =  0
  Width        =  975
End
Begin VB.CommandButton cmdDelete
  Caption      =  "&Delete"
  Height       =  300
  Left         =  1217
  TabIndex     =  8
  Top          =  0
  Width        =  975
End
Begin VB.CommandButton cmdAdd
  Caption      =  "&Add"
  Height       =  300
  Left         =  121
```

```
      TabIndex      =  7
      Top         =  0
      Width        =  975
    End
  End
  Begin VB.Data datPrimaryRS
    Align        =  2  'Align Bottom
    Caption       =  " "
    Connect       =  "Access"
    DatabaseName   =  "C:\ebalance\ebalance.mdb"
    DefaultCursorType=  0  'DefaultCursor
    DefaultType    =  2  'UseODBC
    Exclusive      =  0  'False
    Height        =  345
    Left         =  0
    Options       =  0
    ReadOnly      =  0  'False
    RecordsetType   =  1  'Dynaset
    RecordSource    =  "select * from [Tran Codes]"
    Top          =  1350
    Width         =  5550
  End
```

```
Begin VB.CheckBox chkFields
  DataField     =  "Action"
  DataSource    =  "datPrimaryRS"
  Height        =  285
  Index         =  2
  Left          =  2040
  TabIndex      =  5
  Top           =  700
  Width         =  3375
End
Begin VB.TextBox txtFields
  DataField     =  "Description"
  DataSource    =  "datPrimaryRS"
  Height        =  285
  Index         =  1
  Left          =  2040
  MaxLength     =  30
  TabIndex      =  3
  Top           =  380
  Width         =  3375
End
Begin VB.TextBox txtFields
  DataField     =  "TC"
```

```
      DataSource      =  "datPrimaryRS"
      Height          =  285
      Index           =  0
      Left            =  2040
      TabIndex        =  1
      Top             =  60
      Width           =  735
    End
    Begin VB.Label lblLabels
      Caption         =  "Debit:"
      Height          =  255
      Index           =  2
      Left            =  120
      TabIndex        =  4
      Top             =  700
      Width           =  1815
    End
    Begin VB.Label lblLabels
      Caption         =  "Description:"
```

frmTranCodes - 3

```
      Height          =  255
```

```
    Index       =  1
    Left       =  120
    TabIndex     =  2
    Top        =  380
    Width       =  1815
  End
  Begin VB.Label lblLabels
    Caption      =  "Transaction Code:"
    Height      =  255
    Index       =  0
    Left       =  120
    TabIndex     =  0
    Top        =  60
    Width       =  1815
  End
End
frmUsers - 1



Option Explicit
Function MakeAlpha(AcctName As String)
Dim L%, X%
'Remove all commas
```

```
Do
 L% = InStr(AcctName, ",")
 If L% <> 0 Then Mid$(AcctName, L%) = " "
Loop Until L% = 0

'Remove NickNames "(JIM)"


L% = InStr(AcctName, "(")
If L% <> 0 Then
  X% = InStr(AcctName, ")")
  On Error Resume Next
     If X% <> 0 Then AcctName = Trim$(Left$(AcctName, L% - 1)) & " " &
Trim$(Mid$(AcctN
ame, X% + 1))
  On Error GoTo 0
End If


'Remove Brackets
L% = InStr(AcctName, "[")
If L% <> 0 Then
  X% = InStr(AcctName, "]")
  On Error Resume Next
```

```
      If X% <> 0 Then AcctName = Trim$(Trim$(Left$(AcctName, L% - 1)) & " " &
  Trim$(Mid$
  (AcctName, X% + 1)))
    On Error GoTo 0
  End If


  'Remove 'THE'
  If Left$(AcctName, 4) = "THE " Then AcctName = Mid$(AcctName, 5)


  GoSub RemoveDouble


  BackName:
  'Find and Remove Suffixs
  X% = Len(AcctName)
  Do
    If Mid$(AcctName, X%, 1) = " " Then Exit Do
    X% = X% - 1
  Loop Until X% = 0
  Select Case Mid$(AcctName, X% + 1)
    Case "SR", "JR", "AND", "III", "OR", "MD", "DDS", "DVM", "DBA", "CPA", "II", "IV"
        AcctName = Left$(AcctName, X% - 1)
        GoTo BackName
    Case "SR.", "JR.", "M.D.", "D.D.S.", "D.V.M.", "D.B.A.", "C.P.A."
```

```
        AcctName = Left$(AcctName, X% - 1)
        GoTo BackName
      Case "COMPANY", "CO", "INC", "CLUB", "INC.", "CO.", "ASSOC.", "ASSOC",
    "ASSOCIATES"
        GoSub StripPeroids
        MakeAlpha = Left$(Trim$(AcctName), 12): Exit Function
      Case Else
        GoSub StripPeroids
        X% = Len(AcctName)
        Do
          If Mid$(AcctName, X%, 1) = " " Then Exit Do
          X% = X% - 1
        Loop Until X% = 0
        MakeAlpha = Mid$(AcctName, X% + 1) & ", " + Left$(AcctName, X% - 1)
    End Select
    Exit Function


    StripPeroids:


    frmUsers - 2



    Do
```

```
 L% = InStr(AcctName, ".")
 If L% <> 0 Then Mid$(AcctName, L%) = " ": AcctName = Trim$(AcctName)
Loop Until L% = 0
RemoveDouble:
'Remove Double Spaces
Do
 L% = InStr(AcctName, "  ")
 If L% <> 0 Then AcctName = Left$(AcctName, L%) & Trim$(Mid$(AcctName, L%))
Loop While L% <> 0
Return
End Function


Private Sub cmdAccounts_Click()
CurrentID = txtFields(0)
frmAccounts.Show vbModal, Me
End Sub


Private Sub cmdAdd_Click()
  datPrimaryRS.Recordset.AddNew
  txtFields(0) = datPrimaryRS.Recordset![ID]
  txtFields(3) = Format$(Now, "mm/dd/yyyy")
  txtFields(1).SetFocus
End Sub
```

```
Private Sub cmdDelete_Click()


 If MsgBox("Delete this user?", vbYesNo + vbQuestion, "DELETE USER") <> vbYes
Then E
xit Sub
 Dim SQL$
 SQL$ = "Delete from Accounts where [User ID] =" & txtFields(0)
 dbs.Execute SQL$
 With datPrimaryRS.Recordset
  .Delete
  .MoveNext
  If .EOF Then .MoveLast
 End With
 cmdClose_Click
End Sub



Private Sub cmdUpdate_Click()
On Error Resume Next
 datPrimaryRS.UpdateRecord
If Err Then
```

```
  MsgBox "An error has occured in this update. " & Error$, vbOKOnly, "UPDATE
ERROR"
End If
On Error GoTo 0
datPrimaryRS.Recordset.Bookmark = datPrimaryRS.Recordset.LastModified
txtFields(0) = CurrentID
txtFields(1).SetFocus
Screen.MousePointer = 0
End Sub

Private Sub cmdClose_Click()
 Screen.MousePointer = vbDefault
 Unload Me
End Sub

Private Sub Combo1_Change()


End Sub
frmUsers - 3



Private Sub datPrimaryRS_Error(DataErr As Integer, Response As Integer)
```

```
  'This is where you would put error handling code
  'If you want to ignore errors, comment out the next line
  'If you want to trap them, add code here to handle them
  MsgBox "Data error event hit err:" & Error$(DataErr)
  Response = 0  'Throw away the error
End Sub


Private Sub datPrimaryRS_Reposition()
  Screen.MousePointer = vbDefault
  On Error Resume Next
  'This will display the current record position for dynasets and snapshots
  datPrimaryRS.Caption = "User: " & (datPrimaryRS.Recordset.AbsolutePosition + 1)
End Sub


Private Sub datPrimaryRS_Validate(Action As Integer, Save As Integer)
  'This is where you put validation code
  'This event gets called when the following actions occur
  Select Case Action
    Case vbDataActionMoveFirst
    Case vbDataActionMovePrevious
    Case vbDataActionMoveNext
    Case vbDataActionMoveLast
    Case vbDataActionAddNew
    Case vbDataActionUpdate
    Case vbDataActionDelete
    Case vbDataActionFind
    Case vbDataActionBookmark
    Case vbDataActionClose
      Screen.MousePointer = vbDefault
  End Select
  Screen.MousePointer = vbHourglass
End Sub

Private Sub Form_Load()
Me.Caption = SystemName & " " & Me.Caption
datPrimaryRS.DatabaseName = App.Path & "\eStatement.mdb"
datPrimaryRS.RecordSource = "select * from [Users] Where [ID] = " & CurrentID
datPrimaryRS.Refresh
datPrimaryRS.Recordset.MoveFirst


End Sub


Private Sub Form_Unload(Cancel As Integer)
  Screen.MousePointer = vbDefault
End Sub
```

```
Private Sub txtFields_LostFocus(Index As Integer)
If Index = 1 And Trim$(txtFields(4)) = "" And txtFields(1) <> "" Then
  txtFields(4) = MakeAlpha(txtFields(1))
End If
End Sub
```

```
VERSION 5.00
Begin VB.Form frmUsers
  BorderStyle    =  3  'Fixed Dialog
  Caption       =  "Customer Setup"
  ClientHeight   =  3690
  ClientLeft     =  1095
  ClientTop      =  330
  ClientWidth    =  5550
  Icon          =  (Icon)
  LinkTopic      =  "Form2"
  LockControls    =  -1  'True
  MaxButton      =  0  'False
  MinButton      =  0  'False
  ScaleHeight    =  3690
  ScaleWidth     =  5550
  ShowInTaskbar  =  0  'False
  StartUpPosition =  2  'CenterScreen
  Begin VB.CheckBox chkFields
    DataField      =  "No Ad"
    DataSource     =  "datPrimaryRS"
```

```
      Height         =  285
      Index          =  2
      Left           =  2040
      TabIndex       =  5
      Top            =  2040
      Width          =  3375
   End
   Begin VB.CheckBox chkFields
      DataField      =  "Send Rates"
      DataSource     =  "datPrimaryRS"
      Height         =  285
      Index          =  1
      Left           =  2040
      TabIndex       =  6
      Top            =  2385
      Width          =  3375
   End
   Begin VB.TextBox txtFields
      DataField      =  "Sort Name"
      DataSource     =  "datPrimaryRS"
      Height         =  315
      Index          =  4
      Left           =  2040
   MaxLength      =  50
   TabIndex       =  7
   Top            =  2760
   Width          =  3135
End
Begin VB.CheckBox chkFields
   DataField      =  "Suspend"
   DataSource     =  "datPrimaryRS"
   Height         =  285
   Index          =  0
   Left           =  2040
   TabIndex       =  4
   Top            =  1680
   Width          =  3375
End
Begin VB.TextBox txtFields
   DataField      =  "Date Added"
   DataSource     =  "datPrimaryRS"
   Height         =  315
   Index          =  3
   Left           =  4380
```

```
    MaxLength      =  10
    TabIndex       =  19
    Top            =  0
    Visible        =  0  'False
    Width          =  315
  End
  Begin VB.PictureBox picButtons
    Align          =  2  'Align Bottom
    Appearance     =  0  'Flat
    BorderStyle    =  0  'None
    ForeColor      =  &H80000008&
    Height         =  300
    Left           =  0
    ScaleHeight    =  300
    ScaleWidth     =  5550
    TabIndex       =  15
    Top            =  3045
    Width          =  5550
    Begin VB.CommandButton cmdAccounts
      Caption        =  "&Accounts"
      Height         =  300
      Left           =  3420
  TabIndex      =  21
  TabStop       =  0  'False
  Top           =  0
  Width         =  975
End
Begin VB.CommandButton cmdClose
  Caption       =  "&Close"
  Height        =  300
  Left          =  4505
  TabIndex      =  18
  TabStop       =  0  'False
  Top           =  0
  Width         =  975
End
Begin VB.CommandButton cmdUpdate
  Caption       =  "&Update"
  Height        =  300
  Left          =  2325
  TabIndex      =  8
  Top           =  0
  Width         =  975
End
Begin VB.CommandButton cmdDelete
```

```
        Caption         =   "&Delete"
        Height          =   300
        Left            =   1217
        TabIndex        =   17
        TabStop         =   0   'False
        Top             =   0
        Width           =   975
      End
      Begin VB.CommandButton cmdAdd
        Caption         =   "&Add"
        Height          =   300
        Left            =   121
        TabIndex        =   16
        TabStop         =   0   'False
        Top             =   0
        Width           =   975
      End
   End
   Begin VB.Data datPrimaryRS
frmUsers - 3


      Align           =   2  'Align Bottom
```

```
  Caption      =  " "
  Connect      =  "Access"
  DatabaseName   =  "C:\estatement\estatement.mdb"
  DefaultCursorType=  0  'DefaultCursor
  DefaultType    =  2  'UseODBC
  Exclusive     =  0   'False
  Height       =  345
  Left        =  0
  Options      =  0
  ReadOnly      =  0  'False
  RecordsetType   =  1  'Dynaset
  RecordSource    =  "select * from [Users] Order by [Address Name]"
  Top         =  3345
  Visible      =  0   'False
  Width        =  5550
End
Begin VB.CheckBox chkFields
  DataField     =  "Confirmed"
  DataSource     =  "datPrimaryRS"
  Height       =  285
  Index        =  4
  Left        =  2040
  TabIndex      =  3
```

```
    Top          =  1340
    Width         =  3375
  End
  Begin VB.CheckBox chkFields
    DataField      =  "Charge"
    DataSource      =  "datPrimaryRS"
    Height        =  285
    Index        =  3
    Left         =  2040
    TabIndex        =  2
    Top          =  1020
    Width         =  3375
  End
  Begin VB.TextBox txtFields
    DataField      =  "e-Mail Address"
    DataSource      =  "datPrimaryRS"
    Height        =  285
    Index        =  2
    Left         =  2040
    MaxLength       =  50
    TabIndex        =  1
    Top          =  700
    Width         =  3135
```

```
End
Begin VB.TextBox txtFields
  BackColor    =  &H80000004&
  DataField    =  "ID"
  DataSource   =  "datPrimaryRS"
  Enabled      =  0  'False
  Height       =  285
  Index        =  0
  Left         =  2040
  TabIndex     =  10
  Top          =  60
  Width        =  735
End
Begin VB.TextBox txtFields
  DataField    =  "Address Name"
  DataSource   =  "datPrimaryRS"
  Height       =  315
frmUsers - 4

  Index        =  1
  Left         =  2040
  MaxLength    =  50
```

```
    TabIndex      =  0
    Top           =  380
    Width         =  3135
  End
  Begin VB.Label lblLabels
    Caption       =  "Do not Send Ads:"
    Height        =  255
    Index         =  8
    Left          =  120
    TabIndex      =  24
    Top           =  2040
    Width         =  1815
  End
  Begin VB.Label lblLabels
    Caption       =  "Send C.D. Rates:"
    Height        =  255
    Index         =  7
    Left          =  120
    TabIndex      =  23
    Top           =  2385
    Width         =  1815
  End
  Begin VB.Label lblLabels
```

```
    Caption      =  "Sort Name:"
    Height      =  255
    Index      =  6
    Left      =  120
    TabIndex      =  22
    Top      =  2760
    Width      =  1815
  End
  Begin VB.Label lblLabels
    Caption      =  "Suspend e-Mail:"
    Height      =  255
    Index      =  5
    Left      =  120
    TabIndex      =  20
    Top      =  1680
    Width      =  1815
  End
  Begin VB.Label lblLabels
    Caption      =  "Confirmed:"
    Height      =  255
    Index      =  4
    Left      =  120
    TabIndex      =  14
    Top         =  1340
    Width        =  1815
  End
  Begin VB.Label lblLabels
    Caption       =  "Charge:"
    Height       =  255
    Index        =  3
    Left        =  120
    TabIndex       =  13
    Top         =  1020
    Width        =  1815
  End
  Begin VB.Label lblLabels
    Caption       =  "eMail Address:"
    Height       =  255
frmUsers - 5
```

```
    Index        =  2
    Left        =  120
    TabIndex       =  12
    Top         =  700
    Width        =  1815
```

```
End
Begin VB.Label lblLabels
  Caption      = "Address Name:"
  Height      = 255
  Index      = 1
  Left      = 120
  TabIndex      = 11
  Top      = 380
  Width      = 1815
End
Begin VB.Label lblLabels
  Caption      = "Customer ID:"
  Height      = 255
  Index      = 0
  Left      = 120
  TabIndex      = 9
  Top      = 60
  Width      = 1815
End
End
frmUsersGrid - 1
```

```
Option Explicit



Private Sub cmdAddEdit_Click()
MSFlexGrid1.Col = 0
CurrentID = 0
If MSFlexGrid1.Text <> "" Then CurrentID = Val(MSFlexGrid1.Text)
Dim f As New frmUsers
f.Show vbModal, Me
datPrimaryRS.Refresh
End Sub


Private Sub cmdClose_Click()
  Screen.MousePointer = vbDefault
  Unload Me
End Sub
Private Sub datPrimaryRS_Error(DataErr As Integer, Response As Integer)
  'This is where you would put error handling code
  'If you want to ignore errors, comment out the next line
  'If you want to trap them, add code here to handle them
  MsgBox "Data error event hit err:" & Error$(DataErr)
  Response = 0  'Throw away the error
End Sub
```

```
Private Sub datPrimaryRS_Reposition()
 Screen.MousePointer = vbDefault
 On Error Resume Next
End Sub

Private Sub datPrimaryRS_Validate(Action As Integer, Save As Integer)
 'This is where you put validation code
 'This event gets called when the following actions occur
 Select Case Action
   Case vbDataActionMoveFirst
   Case vbDataActionMovePrevious
   Case vbDataActionMoveNext
   Case vbDataActionMoveLast
   Case vbDataActionAddNew
   Case vbDataActionUpdate
   Case vbDataActionDelete
   Case vbDataActionFind
   Case vbDataActionBookmark
   Case vbDataActionClose
     Screen.MousePointer = vbDefault
 End Select
 Screen.MousePointer = vbHourglass
End Sub
```

```
Private Sub Form_Load()
Me.Caption = SystemName & " " & Me.Caption
datPrimaryRS.DatabaseName = App.Path & "\estatement.mdb"
datPrimaryRS.RecordSource = "Select [ID],[Sort Name] as [Customer Name],[e-Mail
Addre
ss],[Charge] as Chrg,[Confirmed] as Conf,[Suspend] as Susp, [No Ad], [Send Rates] as
Rates,[Date Added] from [Users] Order by [Sort Name]"
MSFlexGrid1.FormatString = "ID   |Name                     |e-Mail Address
                    |Chrg|Conf|Susp|No Ad|Rates|Date Added  "

End Sub

Private Sub Form_Unload(Cancel As Integer)
  Screen.MousePointer = vbDefault
End Sub

Private Sub Form_Resize()
frmUsersGrid - 2

  On Error Resume Next
  'This will resize the grid when the form is resized
  MSFlexGrid1.Height = Me.ScaleHeight - picButtons.Height - 60
```

```
End Sub


Private Sub MSFlexGrid1_DblClick()
cmdAddEdit_Click
End Sub
frmUsersGrid - 1



VERSION 5.00
Object = "{5E9E78A0-531B-11CF-91F6-C2863C385E30}#1.0#0"; "MSFLXGRD.OCX"
Begin VB.Form frmUsersGrid
   Caption         =   "Customers"
   ClientHeight    =   4290
   ClientLeft      =   1110
   ClientTop       =   345
   ClientWidth     =   8175
   Icon            =   (Icon)
   LinkTopic       =   "Form1"
   ScaleHeight     =   4290
   ScaleWidth      =   8175
   StartUpPosition =   2  'CenterScreen
   Begin MSFlexGridLib.MSFlexGrid MSFlexGrid1
      Height          =   3495
```

```
 Left          =  60
 TabIndex      =  2
 Top           =  60
 Width         =  8055
 _ExtentX      =  14208
 _ExtentY      =  6165
 _Version      =  393216
 Cols          =  8
 FixedCols     =  0
 ScrollBars    =  2
 FormatString  =  <...>
End
Begin VB.PictureBox picButtons
  Align        =  2  'Align Bottom
  Appearance   =  0  'Flat
  BorderStyle  =  0  'None
  ForeColor    =  &H80000008&
  Height       =  300
  Left         =  0
  ScaleHeight  =  300
  ScaleWidth   =  8175
  TabIndex     =  0
  Top          =  3645
```

```
        Width          =   8175
        Begin VB.CommandButton cmdAddEdit
          Caption        =   "&Add/Edit"
          Height         =   300
          Left           =   240
          TabIndex       =   3
          Top            =   0
          Width          =   975
        End
        Begin VB.CommandButton cmdClose
          Caption        =   "&Close"
          Height         =   300
          Left           =   1350
          TabIndex       =   1
          Top            =   0
          Width          =   975
        End
      End
      Begin VB.Data datPrimaryRS
        Align          =   2  'Align Bottom
        Caption        =   " "
        Connect        =   "Access"
        DatabaseName   =   "C:\ebalance\ebalance.mdb"
        DefaultCursorType=  0  'DefaultCursor
        DefaultType    =  2  'UseODBC
        Exclusive      =  0   'False
    frmUsersGrid - 2

        Height         =  345
        Left           =  0
        Options        =  0
        ReadOnly       =  0   'False
        RecordsetType  =  1  'Dynaset
        RecordSource   =  ""
        Top            =  3945
        Visible        =  0   'False
        Width          =  8175
      End
    End


    Module1 - 1


    Option Explicit
    Public fMainForm As frmMain
    Public CurrentID As Long, LoanDate As Date, LastUpdate As Date
```

```
Public ReportPath As String, FilePath As String, LoanPath As String
Public ReportSet As String, ReportDateAdd%
Global wrkJet As Workspace
Global dbs As Database
Global UsersDisabled As Boolean
Global tblAccounts As Table
Global tblUsers As Table
Global tblBalances As Table
Global tblTrans As Table
Global tblNSF As Table
Global tblTranCodes As Table
Global tblHistory As Table
Global AutoFlag%, LastComplete As Date, LastTry As Date, NextTry As Date,
FilesMissin
g As Integer, AutoI As Integer
Global tblSetup As Table
Global tblOptions As Table
Global GBalance%, GTrans%, GStatement%, GLoans%, GCSV%, GQwick%,
GMessage$
Global SystemName$, BankName$, StatementFont%, GIFName$
Sub Main()
 Dim fLogin As New frmLogin
 SystemName$ = GetSetting(App.Title, "Settings", "SystemName", "eStatement")
```

```
    BankName$ = GetSetting(App.Title, "Settings", "BankName", "Demo Bank")
    StatementFont% = Val(GetSetting(App.Title, "Settings", "StatementFont", "6"))
    GIFName$ = GetSetting(App.Title, "Settings", "GIFName", "")
    'fLogin.Show vbModal
    'If Not fLogin.OK Then
    '  'Login Failed so exit app
    '  End
    'End If
    'Unload fLogin


    frmSplash.Show
    frmSplash.Refresh


    Set fMainForm = New frmMain
    Load fMainForm
    Unload frmSplash


    fMainForm.Show
End Sub
```

The preferred method may also be described with reference to the process steps as set forth in Figs. 13 through 17. Now, with first reference to Fig. 13, there is displayed the various steps in logical sequence from the beginning of e-statement generation up to the actual building of the e-statements. The statement generation step 168 is initiated by verifying input of the various files described above, for extraction of certain financial data, such as balances, debits and credits to checking accounts, loan accounts and the like, previously described and for ultimate display into the e-statement as shown in Fig. 1. During final verification step 169, a negative response will block continued processing and return to menu reflected in block 170.

If all files are positively verified at 169, all variables are initialized in step 171. The variables initiated in step 171 then are loaded into a format in step 172 and dimension variables 173 are configured. If it is then desired to process the statements at step 174, the functions are run and the statements are made at 175.

The actual creation step 176 for the statements is illustrated in Fig. 14. For the making of the statements 176, the dimension variables 173 are considered and the various source files are searched to confirm their presents at step 177. If the files are not present, step 178, return to run function 175 to step 169. If step 177 confirms the presents of various files, the files are open and the header lines are read as step 179. The lines are then read until the top of the page is identified, step 180. The first 8 lines are read and the account number is extracted at step 181 to confirm account number match up step 182. If the confirmation cannot be made at step 182, step 180 is repeated until confirmation is established. Confirmation of account number through step 182 permits continuance of makeup of the statements and an HTML file is built by reading each line of the statement, step 183. The file creation then is ended, step 184 or steps 180 through 183 repeated until creation of the file. Creation of the file enables return step 185 to process balances in the selected accounts (files) step 186. As part of the processing of the balances 186, the loan files are accessed and processed for balances and the like at 187. The loans are read as a sub-step 188.

The loan sub-step 188 is initiated 189 to process, as shown in Fig. 1, two loan files, trial balances 190. Each line of the text from the files is conducted at 191 and extraction of selected character lines, such as 2 through 11 for purposes of Fig. 1 is effected through step 192. If the characters are numeric, 193 an account number match is effected 194. If the characters are not numeric, 193, steps 191 and 192 are repeated to search for any information which can be utilized to effect an account match-up. If the account number match-up is effected, 194, the balance table is updated with data from 3 file lines, for example, to effect the configuration for Fig. 1 at step 195. The file is now complete, 196 or steps 191 through 195 are repeated until the file is completed and return to main program, 197.

After the return step 197, the trial balances for deposit accounts are read and the balance information is extracted, step 198. If the account does not reflect any active loans, step 188 is not initiated and the deposit trial balances readings is immediately effected.

The deposit trial balances step 198 will now be described. First, old balances from the data base for all accounts are deleted as step 199. Confirmation of the existence of such files is then made at 200. If no files are present, trial balances for deposits sub-step 198 is not effected. If presence of the files is confirmed at 200, the files are processed, for example, 3 as shown in Fig. 1, incrementally at 201. Each file is opened in sequence at 202 and each line is read to find and extract the account number, 203. The account number match-up is either confirmed or not at step 204 and, if not, step 203 is repeated until confirmation of the account number match-up. The balance table is updated at step 205 and the file creation is completed, 206, or steps 203, 204 and 205 are repeated until completion of the file and return to transaction processing, step 207. The processing is continued through step 208 by reading all transactions and non-sufficient funds information sub-step d 209.

The reading of the transactions and NSF information, sub-step 209 is as shown in Fig. 17. The availability of the transactions file is confirmed at 210 and a posting journal is opened, step 211. Each line is read and the account number is extracted, step 212 and an account number match is confirmed 213. If the account number cannot be matched positively, the file is terminated at step 214 or procedure 212 repeated until confirmation is established 213. If the file is terminated at step 214, the run function is continued, step 175 and the statements of processed at 174. If the account number is confirmed at 213, the transaction is added to the transaction table step 215 and creation of the file is terminated, 216, or steps 212, 213 and 215 repeated until file completion. File completion enables run function and statement processing steps 175 and 174, respectively, to be effected. Upon completion of the file 216, the availability of a non-sufficient fund file is checked at step 217 and the NSF report file is opened 218. Each line is read and the account number is extracted 219 with account number match up effected at 220. If there is no match-up, the file is terminated as step 216, or step 219 completed to effect an account match-up. After account match-up, 220, NSF checks are added to the NSF table for generation in the e-statement at step 221 and the file is completed, 222 and the statements are created and processed, steps 175 and 174. The e-statements are built as generally shown in Figs. 2 through 12.

Figs. 17A through 17D illustrate the configuration and orientation of an e-statement printout 300. With particular reference to Fig. 17A, the title block 301 is provided at the upper-most portion of the page with advertising or other special title trailer 302, provided adjacent the title 301. A logo 303 or other artistic embellishment is presented as field 303. A special notice 304 or disclaimer is provided just prior to printout of further specific information. A special printout of the customer number 305 deletes some of the digits to make the customer number incomplete, but is sufficient for the customer to know and identify his account number through his own personal knowledge.

Field 306 designates a time period for coverage of the information within the statement 300. A summary field 307 then is provided which basically summarizes and identifies the various accounts, such as checking 308 and savings 309, also with only partial complete digits of the account numbers for security purposes, 310 and 311. Respective balances 312 and 313 are provided for the accounts. An account activity field 314 serves to identify various debits and credits 315, 316, 317 and 318 for the associated accounts. The summary 307 also includes an ending balance column 319 for printing of the respective ending balances 320 and 321 for the respective accounts 308, 309.

More detail is provided in the statement 300 through a breakout of each of the account numbers identified in the summary 307. As shown, the first account activity summary 322 is for checking account "06", 308. The account number configuration is repeated in field 323 with the previous balance field 324 being used to identify the previous date of information summary and the previous balances identified at 325. A deposit total line 326 is given to identify the number 327 of deposits or other credits and a digitized total of such deposits and other credits provided at 328. Likewise, a debit and withdrawal line 329 provides the total 330 of debits and other withdrawals and a digitized column reflecting such total. Finally, an ending balance 332 includes a field 333 for referencing the date of the ending balance and digitized field 332 for printing out the amount of such ending balance. General summary information for the checking account 308 is provided below line 334. As shown, the account disclosure field 334 may include an identification of average daily balance 335 in numerical format 336 and the total number od days for the statement cycle 337 and indicated as "28" in field 328.

Details of account transactions are identified at 329, such as deposits and other credits 330 identified in a date column 331 together with a description 332 such as deposit 333 or direct deposit 334 or other means. An amount column 335 is provided with digitized amounts identified, such as 336.

Checks are identified in the area 337 by date, columns 338, number 339 and amount 340. A star or asterisk 341 indicates numerical sequence has been broken.

As shown in Fig. 17B, the check transactions are continued and there below various miscellaneous charges of the bank or commercial organization are indicated at 342 by date 343, reference number 344, description 345 and amount 346. These charges may be reflected as a debit card actual debit 347 from a business identified as 348 using a business designation code 349 and physical location 350. Electronic transfers for automatic payment of utilities may be made, such as at 351 to a supplier 352.

A daily balance field 352 is provided for the checking account 308 with a breakdown by date 353 and balance 354, as shown in Figs. 17C.

Also shown in Fig. 17C is a breakdown for another account, in this case, a savings account identified at 353 with a digitized account identifier at 354 with the first 4 digits therein deleted for security purposes. A previous balance line 355 provides the previous balance as of a given date, such as 356, together with total number of deposits or other credits 357 and total of debits and withdrawals 358 and ending balance line is provided 359 and, there below, a field for identifying and calculating the interest earned on a year to date basis through the last payment 360.

General account disclosure information is identified at title 361 which includes an average daily balance disclosure 362, the number of days in the statement cycle 363. Interest earned during the current statement period is provided at 364 together with the annual percentage yield as calculated in a percentage format at 365.

Finally, the general promotional information or advertising is provided at the end of the statement in a general field identified as 366.

Although the invention has been described in terms of specified embodiments which are set forth in detail, it should be understood that this is by illustration only and that the invention is not necessarily limited thereto, since alternative embodiments and operating techniques will become apparent to those skilled in the art in view of the disclosure. Accordingly, modifications are contemplated which can be made without departing from the spirit of the described invention.

What is claimed and desired to be secured by Letters Patent is:

(1) A method incorporating a financial institution computer system for extracting financial data within a data base in the computer system, formatting the data and transmitting the formatted data via electronic mail comprising the steps of:

(a) maintaining electronic information on financial accounts of a customer within said data base in the financial institution computer system;

(b) processing said electronic information within the data base to identify and extract pre-selected data therefrom;

(c) electronically formatting said data for transmission to said customer via electronic mail; and

(d) transmitting the formatted data to a location designated by said customer via electronic mail for storage within and readout on a computer system of said customer.

(2) The method of Claim 1, wherein the step of maintaining electronic information includes the sub-step of creating and maintaining at least one of the following electronic file formats and sortable by individual customer identification code:

(1) checking;

(2) savings;

(3) certificate of deposit;

(4) loan;

(5) discount loan;

(6) simple interest loan;

(7) customer's without account;

(8) posting journal for containing all posted transactions for all account types in account number order;

(9) posting journal for posting transactions for all account types in amount order;

(10) previously transmitted statements file;

(11) laser notice file; and

(12) non-sufficient fund check notice files in officer order.

(3) The method of Claim 2, wherein the step of processing said electronic information includes the sub-steps of generating and creating at least one of the following file formats on a predeterminable time increment basis:

(1) mini trial balance for each checking account;

(2) mini trial balance for each savings account;

(3) mini trial balance for each certificate of deposit account;

(4) mini trial balance for each loan account;

(5) mini trial balance for each discount loan account;

(6) trial balance for each simple interest loan account;

(7) trial balance for each customer's without account;

(8) all posted transactions for all account types in account numerical order;

(9) posted transactions for all account types in amount order;

(10) all statements printed for a previous statement transmission period;

(11) a notice in laser format; and

(12) all NSF checks in officer order.

(4) The method of Claim 1 wherein the step of electronically formatting said data includes the sub-step of creating within the financial institution computer system a main menu for said data including sub-menu selections for customers, options, verify files, enable auto e-mail, generate e-mail, broadcast, help and exit.

(5) The method of Claim 1 wherein said step of electronically formatting said data for transmission to said customer includes the sub-step of creating within said financial institution computer system a main menu selection of customers to whom the formatted data is to be transmitted.

(6) The method of Claim 5, further including the sub-step of generating an identification symbol in numerical format for each said customer and including entry of an e-mail address for each such identified customer.

(7) The method of Claim 4 further comprising the sub-step of including within the customer menu an indicator of charge/no charge to the customer for transmission of said formatted data.

(8) The method of Claim 4 further comprising the sub-step of including within the customer menu an indicator of confirmation/no confirmation by the customer of the e-mail address indicated in said customer menu.

(9) The method of Claim 4 further comprising the sub-step of actuating means indicated in the customer main menu for suspending formatted data transmissions to a customer.

(10) The method of Claim 4 further comprising the sub-step of activating means prohibiting advertising being sent in the data transmission to the customer.

(11) The method of Claim 4 further comprising the sub-step of activating means for generating an interest rate calculation in an attachment to the customer with the data transmission on a pre-determined time basis.

(12) The method of Claim 4 further comprising the sub-step of activating the add/edit field in the customer main menu whereby at least one of the following fields are completed or changed:

(1) customer name;

(2) e-mail address;

(3) charge designation;

(4) confirmation of e-mail address by customer;

(5) suspend e-mail transmission;

(6) advertisement transmission block;

(7) transmission of interest rates on designated accounts; and

(8) name of customer by sort designation.

(13) The method of Claim 1 wherein the step of processing said electronic information includes the sub-step of generating a customer set up configuration by sorting the data field for each customer identification data reflected in at least one of the following fields:

(1) account number;

(2) reference number;

(3) account name;

(4) account type;

(5) balance for designated account;

(6) transactions for designated account;

(7) non-sufficient funds designation; and

(8) day or date for report transmission.

(14) The method of Claim 1 wherein the step of processing said electronic information includes the sub-step of enabling a statement automatic set-up mode to generate separate, individual processing attempts, spaced apart by time designations subsequent to a real time selection for initiating the processing step.

(15) The method of Claim 1 wherein the step of electronically formatting said data includes the sub-step of electronically generating at least one of the following financial data:

(1) account balance;

(2) account transactions;

(3) loan data;

(4) statement attachments;

(5) CSV attachments; and

(6) Qwicken attachments.

(16) The method of Claim 1 wherein the step of processing said electronic information includes the sub-step of verifying availability and access to designated files within the electronic information required for formatting said data for transmission to said customer.

(17) The method of Claim 1 wherein the step of formatting said data for transmission includes the sub-step of providing in broadcast mode to each customer a manually generated message for inclusion in and transmission of the formatted data.

(18) A system for notifying a customer of financial information, comprising:

(a) a computer system including a data base including information on customer financial accounts and wherein the financial accounts are maintained in a financial institution computer program; and

(b) program logic implemented in the financial institution computer program comprising:

(1) means for maintaining electronic information within said program on the financial accounts;

(2) means for processing said electronic information to identify and extract the pre-selected data therefrom; and

(3) means for electronically formatting said data for transmission to said customer via electronic mail; and

(4) means for transmitting the formatted data to a location designated by a customer via electronic mail for storage within and readout on a customer computer system.

(19) The system of Claim 18 wherein the program logic further includes means for creating and maintaining at least one of the following electronic file formats sortable by individual customer identification code.

(1) checking;

(2) savings;

(3) certificate of deposit;

(4) loan;

(5) discount loan;

(6) simple interest loan;

(7) customer's without account;

(8) posting journal for containing all posted transactions for all account types in account number order;

(9) posting journal for posting transactions for all account types in amount order;

(10) previously transmitted statements file;

(11) laser notice file; and

(12) non-sufficient fund check notice files in officer order.

(20) The system of Claim 18 wherein the means for processing said electronic information includes means for generating and creating at least one of the following file formats on a predeterminable time increment basis:

(1) mini trial balance for each checking account;

(2) mini trial balance for each savings account;

(3) mini trial balance for each certificate of deposit account;

(4) mini trial balance for each loan account;

(5) mini trial balance for each discount loan account;

(6) trial balance for each simple interest loan account;

(7) trial balance for each customer's without account;

(8) all posted transactions for all account types in account numerical order;

(9) posted transactions for all account types in amount order;

(10) all statements printed for a previous statement transmission period;

(11) a notice in laser format; and

(12) all NSF checks in officer order.

(21) The system of Claim 18 wherein the means for electronically formatting said data includes means for creating within the financial institution computer program a main menu for said data including sub-menu selections for customers, options, verify files, enable auto e-mail, generate e-mail, broadcast, help and exit.

(22) The system of Claim 18 wherein the means for electronically formatting said data further includes means for creating within the said financial institution computer program a main menu selection of customers to whom the formatted data is to be transmitted.

(23) The system of Claim 18 wherein the program logic further includes means for generating an identification symbol in numerical format for each said customer and including entry of an e-mail address for each such identified customer.

(24) The system of Claim 18 wherein the program logic further includes means for indicating a charge or no charge to the customer for transmission of said formatted data.

(25) The system of Claim 18 wherein said program logic further includes means for receipt of electronic confirmation by the customer of the e-mail address indicated by the customer.

(26) The system of Claim 18 further including program logic including means for suspending formatted data transmissions to a customer.

(27) The system of Claim 18 wherein the program logic further includes means for prohibiting the electronic transmission to the customer of advertising with the formatted data.

(28) The system of Claim 18 wherein the program logic further includes means for generating an interest rate calculation in an attachment to the customer with the data transmission on a pre-determined time basis.

(29) The system of Claim 18 (the program logic further includes means for activating an add/edit field in a customer main menu whereby at least one of the following fields may be completed or changed:

(1) customer name;

(2) e-mail address;

(3) charge designation;

(4) confirmation of e-mail address by customer;

(5) suspend e-mail transmission;

(6) advertisement transmission block;

(7) transmission of interest rates on designated accounts; and

(8) name of customer by sort designation.

(30) The system of Claim 18 wherein the program logic further includes means for sorting data for each customer reflected in at least one of the following fields:

(1) account number;

(2) reference number;

(3) account name;

(4) account type;

(5) balance for designated account;

(6) transactions for designated account;

(7) non-sufficient funds designation; and

(8) day or date for report transmission.

(31) The system of Claim 18 wherein the program logic further includes means for enabling a statement automatic set up mode to generate separate, individual processing attempts, spaced apart by time designations subsequent to a real time selection for initiating the process step.

(32) The system of Claim 18 wherein the program logic further includes means for generating at least one of the following financial data:

(1) account balance;

(2) account transactions;

(3) loan data;

(4) statement attachments;

(5) CSV attachments; and

(6) Qwicken attachments.

(33) The system of Claim 18 wherein the program logic further includes means for verifying availability and access to designated files within the computer system required for formatting said data for transmission to said computer.

(34) The system of Claim 18 wherein the program logic includes means for providing in broadcast mode to each customer a manually generated message for inclusion in and transmission of the formatted data.

(35) An article of manufacture for use in programming a financial institution computer system maintaining a data base including information on a plurality of customer accounts, the article of manufacture comprising a computer useable storage medium having at least one computer program stored therein that causes a financial institution computer system to perform steps of:

(a) maintaining electronic information in the financial institution computer system on financial accounts;

(b) processing said electronic information to identify and abstract preselected data therefrom;

(c) electronically formatting said data for transmission to said customer via electronic mail; and

(d) transmitting the formatted data to a location designated by a customer via electronic mail for storage within and readout on a customer computer system.

(36) The article manufacture of Claim 35 wherein the step of maintaining electronic information includes as sub-step of creating and maintaining at least one of

the following electronic file formats sortable by individual customer identification code:

(1) checking;

(2) savings;

(3) certificate of deposit;

(4) loan;

(5) discount loan;

(6) simple interest loan;

(7) customer's without account;

(8) posting journal for containing all posted transactions for all account types in account number order;

(9) posting journal for posting transactions for all account types in amount order;

(10) previously transmitted statements file;

(11) laser notice file; and

(12) non-sufficient fund check notice files in officer order.

(37) The article manufacture of Claim 35 wherein the step of processing said electronic information includes as sub-step of generating and creating at least one of the following file formats on a pre-determinable time increment basis:

(1) mini trial balance for each checking account;

(2) mini trial balance for each savings account;

(3) mini trial balance for each certificate of deposit account;

(4) mini trial balance for each loan account;

(5) mini trial balance for each discount loan account;

(6) trial balance for each simple interest loan account;

(7) trial balance for each customer's without account;

(8) all posted transactions for all account types in account numerical order;

(9) posted transactions for all account types in amount order;

(10) all statements printed for a previous statement transmission period;

(11) a notice in laser format; and

(12) all NSF checks in officer order.

(38) The article manufacture of Claim 35 wherein the step of electronically formatting said data includes the sub-step of creating within the financial institution computer system a main menu for said data including sub-menu selections for customers, options, verify files, enable auto e-mail, generate e-mail, broadcast, help and exit.

(39) The article manufacture of Claim 35 wherein the step of electronically formatting said data for transmission to said customer includes the sub-step of creating within said financial institution computer system a main menu selection of customers to whom the formatted data is to be transmitted.

(40) The article of manufacture of Claim 35 further including the sub-step of generating an identification symbol in numerical format for each said customer and including entry of an e-mail address for each identified customer.

(41) The article of manufacture of Claim 35 further comprising the sub-step of including within the customer menu an indicator of charge/no-charge to the customer for transmission of said formatted data.

(42) The article of manufacture of Claim 35 further comprising the sub-step of including an indicator of confirmation/no confirmation by the customer of the e-mail address indicated in the program logic.

(43) The article of manufacture of Claim 35 further comprising the step of activating means indicated in the program logic for suspending formatted data transmissions to a client.

(44) The article of manufacture of Claim 35 further comprising the step of activating means prohibiting advertising being sent in the data transmission to the customer.

(45) The article of manufacture of Claim 35 further comprising the step of activating means for generating and interest rate calculation in an attachment to the customer with the data transmission on a pre-determined time basis.

(46) The article of manufacture of Claim 35 further comprising the step of activating an add/edit field in the program logic whereby at least one of the following fields are completed or changed:

(1) customer name;

(2) e-mail address;

(3) charge designation;

(4) confirmation of e-mail address by customer;

(5) suspend e-mail transmission;

(6) advertisement transmission block;

(7) transmission of interest rates on designated accounts; and

(8) name of customer by sort designation.

(47) The article of manufacture of Claim 35 further comprising the step of generating a customer set up configuration by sorting a data field for each customer identification data reflected in at least one of the following fields:

(1) account number;

(2) reference number;

(3) account name;

(4) account type;

(5) balance for designated account;

(6) transactions for designated account;

(7) non-sufficient funds designation; and

(8) day or date for report transmission.

(48) The article of manufacture of Claim 35 wherein the step of electronically formatting the data includes the step of electronically generating at least of the following financial data:

(1) account balance;

(2) account transactions;

(3) loan data;

(4) statement attachments;

(5) CSV attachments; and

(6) Qwicken attachments.

(49) The article of manufacture of Claim 35 further including the step of processing the electronic information and enabling a statement automatic set-up mode to generate separate, individual processing attempts, spaced apart by time designations subsequent to a real time selection for initiating the processing step.

(50) The article of manufacture of Claim 35 wherein at least one of the following financial data files is sourced and data extracted therefrom:

(1) account balance;

(2) account transactions;

(3) loan data;

(4) statement attachments;

(5) CSV attachments; and

(6) Qwicken attachments.

(51) The article of manufacture of Claim 35 wherein the step of processing includes the sub-step of verifying availability and access to designated files within the electronic information required for formatting the data for transmission to the customer.

(52) The article of manufacture of Claim 35 wherein the step of formatting the data for transmission includes the sub-step of providing in broadcast mode to each customer a manually generated message for inclusion in and transmission of the formatted data.